

ĐẠI HỌC BÁCH KHOA HÀ NỘI



ĐỒ ÁN TỐT NGHIỆP

THIẾT KẾ THIẾT BỊ ĐỌC GHI MÀN HÌNH
CỘT BƠM XĂNG DẦU

NGUYỄN THÁI SƠN

son.nt212951@sis.hust.edu.vn

Ngành Kỹ thuật Điều khiển và Tự động hóa

Giảng viên hướng dẫn: PGS.TS. Nguyễn Quốc Cường

Chữ ký của GVHD

Khoa: Tự động hóa

Trường: Điện - Điện tử

Hà Nội, 3/2025

**NHIỆM VỤ
ĐỒ ÁN TỐT NGHIỆP**

Họ và tên sinh viên: Nguyễn Thái Sơn

Khóa: K66

Trường: Điện – Điện tử

Ngành: KT ĐK & TĐH

1. Tên đề tài

Thiết kế thiết bị đọc ghi màn hình cột bơm xăng dầu

2. Nội dung đề tài

Thiết kế và triển khai hệ thống gồm phần cứng và phần mềm để thu dữ liệu hiển thị trên màn hình cây xăng, giải mã và lưu các lượt bơm xăng. Lấy mẫu thông tin màn hình với tần số 100Hz. Có kết nối mạng LAN với máy tính nội bộ. Thiết bị có giao diện giám sát và điều khiển dạng Desktop App.

Các công việc bao gồm:

- Thiết kế mạch thu dữ liệu màn hình.
- Thu thập và phân tích tín hiệu gửi tới màn hình.
- Thiết kế và triển khai firmware cho thiết bị thu dữ liệu màn hình. Cụ thể là lập trình cho vi điều khiển STM32 tại khối xử lý trung tâm.
- Thiết kế và triển khai phần mềm điều khiển và giải mã. Cụ thể là viết phần mềm trên máy tính thực hiện giải mã dữ liệu màn hình, phát hiện phiên bơm và điều khiển relay cho thiết bị.
- Triển khai cơ sở dữ liệu và giao diện hiển thị dữ liệu màn hình.
- Chạy kiểm thử hệ thống tại hiện trường.

3. Thời gian giao đề tài: 18/02/2025

4. Thời gian hoàn thành: 18/6/2025

Ngày 18 tháng 6 năm 2025
CÁN BỘ HƯỚNG DẪN

Nguyễn Quốc Cường

LỜI CẢM ƠN

Đây là mục tùy chọn, nên viết phần cảm ơn ngắn gọn, tránh dùng các từ sáo rỗng.

Hà Nội, ngày 24 tháng 03 năm 2025

Sinh viên thực hiện

Nguyễn Thái Sơn

TÓM TẮT ĐỒ ÁN

(Sẽ được bổ sung vào giai đoạn cuối của đồ án)

MỤC LỤC

DANH MỤC KÝ HIỆU VÀ CHỮ VIẾT TẮT	i
DANH MỤC HÌNH VẼ	iii
DANH MỤC BẢNG BIỂU	iv
CHƯƠNG 1. GIỚI THIỆU CHUNG	1
1.1 Vai trò của thiết bị đọc ghi màn hình cột bơm xăng dầu	1
1.2 Khái quát về yêu cầu và chức năng của hệ thống	1
1.3 Phạm vi nghiên cứu	2
1.4 Phương pháp nghiên cứu	3
1.5 Cấu trúc đồ án	4
1.6 Kết luận chương	4
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT	5
2.1	5
CHƯƠNG 3. THIẾT KẾ HỆ THỐNG	6
3.1 Phân tích yêu cầu hệ thống	6
3.1.1 Thiết bị đọc ghi màn hình	6
3.1.2 Phần mềm giải mã	6
3.1.3 Hệ thống mạng	7
3.2 Mô hình thiết kế tổng thể	7
3.2.1 Sơ đồ khối chức năng thiết bị đọc ghi màn hình (Device) . . .	7
3.2.2 Sơ đồ khối chức năng phần mềm giải mã (Device Service) . .	8
3.3 Thiết kế phần cứng thiết bị đọc màn hình	9
3.3.1 Sơ đồ khối phần cứng thiết bị đọc màn hình	9
3.3.2 Lựa chọn linh kiện và vẽ sơ đồ nguyên lý	9
3.4 Thiết kế firmware thiết bị đọc màn hình	14
3.4.1 Kiến trúc triển khai firmware	14
3.4.2 Trình tự giao tiếp giữa các khối	15
3.5 Thiết kế phần mềm giải mã Device Service	18
3.5.1 Kiến trúc triển khai phần mềm	18
3.5.2 Trình tự giao tiếp và thuật toán cho các khối	19

3.6 Giao diện điều khiển cho người dùng	22
3.7 Tổng kết chương	24
CHƯƠNG 4. MÔ PHỎNG VÀ KẾT QUẢ	25
4.1 Quy trình triển khai	25
4.1.1 Mạch phần cứng thiết bị đọc màn hình	25
4.1.2 Firmware thiết bị đọc màn hình	26
4.1.3 Phần mềm giải mã Device Service	32
4.1.4 Viết phần mềm Client UI hiển thị các phiên borm	40
KẾT LUẬN	44
TÀI LIỆU THAM KHẢO	45
PHỤ LỤC	46
A Một số phương pháp đo và hiệu chuẩn	46

DANH MỤC KÝ HIỆU VÀ CHỮ VIẾT TẮT

HESS	Hybrid Energy Storage System
SC	Super Capacitor
EMS	Energy Management Strategy

DANH MỤC HÌNH VẼ

Hình 1.1. Minh họa mục tiêu đầu ra của hệ thống	1
Hình 1.2. Mạch phần cứng màn hình cây xăng hãng ZCheng	4
Hình 3.1. Các thành phần của hệ thống	6
Hình 3.2. Sơ đồ khái niệm chức năng thiết bị đọc ghi màn hình (Device)	7
Hình 3.3. Sơ đồ khái niệm chức năng phần mềm giải mã và chốt phiên bơm (Device Service)	8
Hình 3.4. Sơ đồ khái triển khai phần cứng thiết bị đọc ghi màn hình	9
Hình 3.5. Sơ đồ nguyên lý khái xử lý trung tâm	11
Hình 3.6. Sơ đồ nguyên lý khái đầu vào tín hiệu	12
Hình 3.7. Sơ đồ nguyên lý khái nguồn và giao tiếp Ethernet	13
Hình 3.8. Sơ đồ khái triển khai firmware cho thiết bị	14
Hình 3.9. Trình tự giao tiếp: Thiết lập kết nối tới phần mềm giải mã Device Service	15
Hình 3.10. Trình tự giao tiếp: thu thập và đẩy dữ liệu màn hình	16
Hình 3.11. Trình tự giao tiếp: cập nhật firmware từ xa (OTA)	17
Hình 3.12. Sơ đồ khái triển khai phần mềm Device Service	18
Hình 3.13. Trình tự giao tiếp cho phần mềm giải mã: Giải mã màn hình	20
Hình 3.14. Trình tự giao tiếp phần mềm trong trường hợp điều khiển relay	21
Hình 3.15. Trình tự giao tiếp phần mềm trong trường hợp thực hiện OTA	22
Hình 3.16. Sơ đồ khái triển khai phần mềm giao diện	23
Hình 3.17. Giao tiếp giữa các process của ứng dụng thông qua IPC	24
Hình 4.1. Mạch PCB - Mặt trên	25
Hình 4.2. Mạch PCB - Mặt ngang và mặt dưới	25
Hình 4.3. Sản phẩm thiết bị đọc ghi màn hình cột bơm: mặt trên	26
Hình 4.4. Sản phẩm thiết bị đọc ghi màn hình cột bơm: mặt ngang và dưới	26
Hình 4.5. Triển khai: cấu hình các chân trong STM32CubeIDE	27
Hình 4.6. Thuật toán xử lý ngắn nhận dữ liệu màn hình	28
Hình 4.7. Thuật toán xử lý ngắn nhận dữ yêu cầu Device Service	29
Hình 4.8. Thuật toán cho chương trình chính	30
Hình 4.9. :Tạo C++ Win32 Project trên Visual Studio	32
Hình 4.10. Thuật toán giải mã màn hình và phát hiện lượt bơm	34
Hình 4.11. Các trường dữ liệu trong một Log hoặc Sublog	35
Hình 4.12. Sơ đồ trạng thái cho cột bơm	36

Hình 4.13. Thuật toán thực hiện OTA trên Device Service	38
Hình 4.14. Device Service: Các file logs	40
Hình 4.15. Mẫu dự án sử dụng Electron + Vite + ReactJS	41
Hình 4.16. Các trường lưu trong bảng Log của Database	42
Hình 4.17. Giao diện hiển thị Client UI	43

DANH MỤC BẢNG BIỂU

Bảng 0.1. Bảng cập nhật báo cáo.	v
Bảng 0.2. Bảng kế hoạch dự án.	vi
Bảng 0.2. Bảng kế hoạch dự án.	vii
Bảng 0.3. Biên bản cuộc họp.	viii
Bảng 4.1. Cấu trúc gói tin giữa Device và Device Service	31
Bảng 4.2. Danh sách các loại CMD trong enum Cmd_t	32
Bảng 4.3. Các loại sự kiện màn hình	35
Bảng 4.4. Mô tả các trạng thái máy (Machine State)	36
Bảng 4.5. Callback được gọi sau khi xác định trạng thái mới	37
Bảng 4.6. Danh sách các message_type của Device Service	39

Bảng cập nhật báo cáo

Bảng 0.1. Bảng cập nhật báo cáo.

Ngày	Nội dung báo cáo	Sửa đổi / ghi chú
4/3	Bảng kế hoạch	Lập bảng kế hoạch công việc
24/3	Chương 1: Tổng quan hệ thống	Viết nội dung tổng quan hệ thống
25/3	Meeting note	Ghi các meeting note vào báo cáo

Kế hoạch thực hiện

Bảng 0.2. Bảng kế hoạch dự án.

Tuần	Nhiệm vụ	Yêu cầu cần đạt	Trạng thái
26	Xác định mục tiêu đề tài	Liệt kê yêu cầu bài toán, chức năng thiết bị	Hoàn thành
27	Tìm hiểu đặc điểm cột bơm, xác định phạm vi bài toán	Báo cáo trình bày đặc điểm cột bơm	Hoàn thành
28	Lập bảng kế hoạch công việc	File Excel có bảng liệt kê các đầu việc chi tiết cho các tuần cùng với tiến độ được cập nhật	Hoàn thành
29	Vẽ sơ đồ hệ thống	Sơ đồ tổng quan hệ thống	Hoàn thành
30	Liệt kê và tìm hiểu các công nghệ sử dụng	Báo cáo tìm hiểu các phần tử IC được sử dụng, CPLD và STM32	Hoàn thành
31	Tìm hiểu mạch cứng	Đọc hiểu và viết tài liệu về mạch cứng, tìm hiểu các linh kiện	Đang thực hiện
32	Vẽ thiết kế mạch cứng	Bản thiết kế mạch cứng	Chưa làm
32	Triển khai hệ thống phần mềm	Xây dựng kiến trúc source code và chuẩn API	Chưa làm
33	Triển khai backend server	Demo backend server giao tiếp bằng chuẩn API đã đưa ra	Chưa làm
34	Triển khai giao diện hiển thị	Demo giao diện mô phỏng màn hình cây xăng theo thời gian thực	Chưa làm
35	Kiểm thử, chạy thử hệ thống và quay video demo	Demo chạy thử toàn hệ thống, trình bày các lỗi phát sinh nếu có	Chưa làm

Bảng 0.2. Bảng kế hoạch dự án.

Tuần	Nhiệm vụ	Yêu cầu cần đạt	Trạng thái
36	Viết báo cáo quyển	Trình bày quyển	Chưa làm
37	Viết báo cáo dạng báo cáo khoa học	Trình bày báo cáo	Chưa làm

Biên bản cuộc họp

Bảng 0.3. Biên bản cuộc họp.

Ngày	Nội dung	Quyết định	Nhiệm vụ tiếp theo
25/2	Giao đề tài và nêu ý tưởng chung về đề tài	Đưa ra các đầu việc, thống nhất về đầu mối	Tìm hiểu công nghệ, dựng hệ thống và tài liệu hóa
4/3	Trình bày sơ bộ về tiến độ, bảng kế hoạch (đã đưa và đang chờ duyệt). Thông nhất lịch họp hàng tuần	Khoảng hết tuần 5, đầu tuần 6 báo cáo overview và kế hoạch đồ án	Bắt đầu viết tài liệu hệ thống

CHƯƠNG 1. GIỚI THIỆU CHUNG

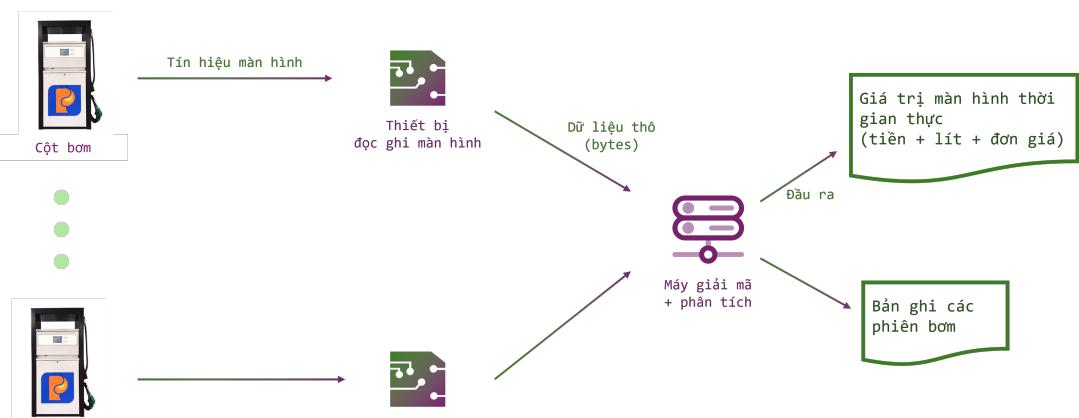
1.1 Vai trò của thiết bị đọc ghi màn hình cột bơm xăng dầu

Công nghệ số mang đến ngày càng nhiều lợi ích to lớn về mặt tiện dụng, hiệu quả và tiết kiệm thời gian cho doanh nghiệp cũng như người dùng. Từ đó, nhu cầu lớn về chuyển đổi số được phát sinh. Các thông tin giao dịch, hóa đơn trên giấy tờ ngày càng được thay thế bằng các giao dịch, hóa đơn điện tử, thuận tiện cho truyền tải thông tin, lưu trữ và quản lý. Các hệ thống trạm xăng và cột bơm cũng không phải ngoại lệ với nhu cầu đó, các giao dịch, lượt bơm xăng cũng cần được ghi lại dưới dạng dữ liệu số để gửi lên máy chủ của doanh nghiệp, thuận tiện lưu trữ và thống kê một cách chính xác, ổn định và giảm hoặc triệt tiêu những lỗi con người có thể gây ra.

Những cột bơm mới được bán gần đây cũng đã được sản xuất tích hợp chức năng xuất hóa đơn điện tử. Nhưng còn một lượng rất lớn những cây xăng với công nghệ cũ vẫn còn đạt tiêu chuẩn và sử dụng tốt được ở nhiều trạm bơm trên khắp cả nước, việc bỏ hẳn những cây xăng này đi thay bằng những cây xăng công nghệ mới sẽ tốn nhiều thời gian và chi phí đầu tư, gây lãng phí cũng như thải nhiều rác thải điện tử hơn cho môi trường. Từ đó xuất hiện nhu cầu cải tiến những cây xăng cũ để xuất được hóa đơn điện tử, chi phí và thời gian doanh nghiệp bỏ ra sẽ được tối ưu hơn.

Từ yêu cầu trực tiếp của doanh nghiệp, báo cáo này được viết để đưa ra phương án thiết kế và triển khai hệ thống có thể đọc ghi và giải mã, phân tích dữ liệu màn hình cây xăng, từ đó phát hiện được các lần giao dịch và lưu trữ, quản lý các lượt giao dịch - lượt bơm xăng đó.

1.2 Khái quát về yêu cầu và chức năng của hệ thống



Hình 1.1. Minh họa mục tiêu đầu ra của hệ thống

Mục tiêu là thiết kế thiết bị đọc ghi và giải mã màn hình cây xăng đáp ứng những nhu cầu sau:

1. Giám sát số liệu hiển thị trên cột bơm theo thời gian thực

Hệ thống có thể đọc và hiển thị trực tiếp dữ liệu cột bơm theo thời gian thực bao gồm: số tiền, số lít và đơn giá. Người dùng có thể theo dõi trạng thái hiển thị của cột bơm ngay tại phòng điều khiển mà không cần quan sát trực tiếp màn hình cột.

2. Phát hiện các lượt bơm hoàn chỉnh

Thông qua các số liệu thời gian thực (số tiền, số lít, đơn giá) của cột bơm, hệ thống phát hiện được đâu là bắt đầu và kết thúc của một lượt bơm xăng, thông báo một lượt bơm/hóa đơn hoàn chỉnh.

3. Lưu trữ và hiển thị các lượt bơm

Các lượt bơm được lưu trữ trong cơ sở dữ liệu, tạo cơ sở để có thể xuất các hóa đơn điện tử.

4. Hỗ trợ điều khiển từ xa

Việc ghi dữ liệu cột bơm có thể được điều khiển bật tắt từ xa, thông qua giao diện phần mềm.

5. Cập nhật phần mềm từ xa (OTA)

Hỗ trợ cập nhật phần mềm từ xa, cho phép người dùng cập nhật phiên bản mới từ giao diện người dùng để có tối ưu những tính năng, tiện ích mới nhất từ phiên bản phần cứng đã có.

Thông qua việc triển khai mục tiêu trên, đề tài hướng tới tạo ra một hệ thống đọc và giải mã màn hình cây xăng hoàn chỉnh, đáp ứng nhu cầu quan sát từ xa các cột bơm theo thời gian thực, tự động phát hiện và ghi lại được các lượt bơm, tạo tiền đề để phát triển các ứng dụng như là xuất hóa đơn điện tử, thanh toán không chạm, và các ứng dụng khác.

1.3 Phạm vi nghiên cứu

Thiết bị đọc ghi màn hình cột bơm xăng dầu có thể đọc ghi và giải mã loại cột bơm truyền thống có đặc điểm sau:

- Hãng sản xuất: ZCheng
- Loại cáp màn hình: 8P 3.96mm
- Số lượng chân cáp: 8
- Số chân tín hiệu của cáp: 3
- Giao thức: SPI
- Độ lớn frame: 22 byte
- Bit rate: 400Kb/s
- Tần số frame: 100Hz

Đây là loại cột bơm cũ phổ biến tại các trạm xăng trên thị trường. Phạm vi chức năng của hệ thống bao gồm:

1. Giám sát số liệu hiển thị trên cột bơm theo thời gian thực

Thu và giải mã dữ liệu SPI gửi tới màn hình thiết bị. Hiển thị giá trị màn hình (số tiền, số lít, đơn giá) theo thời gian thực trên giao diện người dùng.

2. Phát hiện các phiên bơm hoàn chỉnh

Sau khi bơm xăng, người vận hành dừng bơm một thời gian (5s) hoặc nhấn nút reset để bắt đầu phiên bơm mới, thiết bị phát hiện và ghi lại giá trị màn hình hiện tại thành một phiên bơm.

3. Lưu trữ và hiển thị các lượt bơm

Các bản ghi phiên bơm được lưu vào cơ sở dữ liệu ngay trên máy tính nội bộ và hiển thị lên giao diện quản lý

4. Hỗ trợ điều khiển từ xa

Thiết bị đọc màn hình cây xăng có 2 relay đóng ngắn có thể nối tiếp vào cột bơm. Từ giao diện quản lý, có thể điều khiển relay đóng ngắn bằng nút ấn trên giao diện.

5. Cập nhật phần mềm từ xa (OTA)

Từ giao diện quản lý, có thể tải (upload) file phần mềm dạng mã nhị phân, gửi cho thiết bị đọc màn hình để thiết bị tự cập nhật.

1.4 Phương pháp nghiên cứu

Bối cảnh màn hình cây xăng là các màn hình cũ, không có tài liệu cụ thể về thiết phần cứng và giao thức.

Do đó, cần khảo sát tín hiệu màn hình. Sử dụng mạch thu và phần mềm của Logic Analyzer để thu tín hiệu theo các bước sau:

- Sử dụng đồng hồ đo áp, xác định các chân nguồn, đất, tín hiệu và các chân có điện áp cao (để tránh cắm chân Logic Analyzer vào các chân có điện áp cao này).

- Sử dụng mạch cứng và phần mềm Logic Analyzer nối vào các chân tín hiệu tới màn hình để thu các tín hiệu. Đồng thời, sử dụng máy quay quay lại màn hình hiển thị trong quá trình thu.

- Thực hiện các thao tác phổ biến: reset màn hình, bơm xăng, preset màn hình (đặt trước giá trị tiền hoặc lít cần bơm) rồi ghi lại tín hiệu và video màn hình trong quá trình vận hành thao tác.

- Khảo sát tín hiệu, so sánh tín hiệu với giá trị hiển thị trên màn hình để xác định:

- Vị trí các chân (nguồn, đất, cấp xung, dữ liệu, ...)
- Giao thức sử dụng (SPI), dấu hiệu bắt đầu và kết thúc một gói tin
- Ý nghĩa các byte trong gói tin

Từ đó, chế tạo mạch thu gói tin theo giao thức SPI và thiết kế phần mềm giải mã.



Hình 1.2. Mạch phần cứng màn hình cây xăng hãng ZCheng

1.5 Cấu trúc đồ án

Nội dung được trình bày ở các chương tiếp theo bao gồm:

- Tổng quan lý thuyết và công nghệ: Trình bày các nền tảng lý thuyết và mô tả các công nghệ được sử dụng trong thiết kế.
- Thiết kế hệ thống sản phẩm: Bao gồm sơ đồ các khôi chức năng của thiết bị đọc ghi màn hình và phần mềm giải mã, cùng với thiết kế chi tiết (mạch cứng và phần mềm) từng khôi, trình tự giao tiếp giữa các khôi.
- Triển khai và thử nghiệm: Thử nghiệm thiết bị trên màn hình cột bơm tại hiện trường, đánh giá kết quả.
- Kết luận và hướng phát triển.

1.6 Kết luận chương

Trong chương này, em đã đưa ra bối cảnh thực tế của bài toán, giải thích nhu cầu thực tế của thiết bị đọc ghi màn hình cây xăng, cùng với lợi ích mà nó mang lại trong quá trình phát triển và chuyển đổi số.

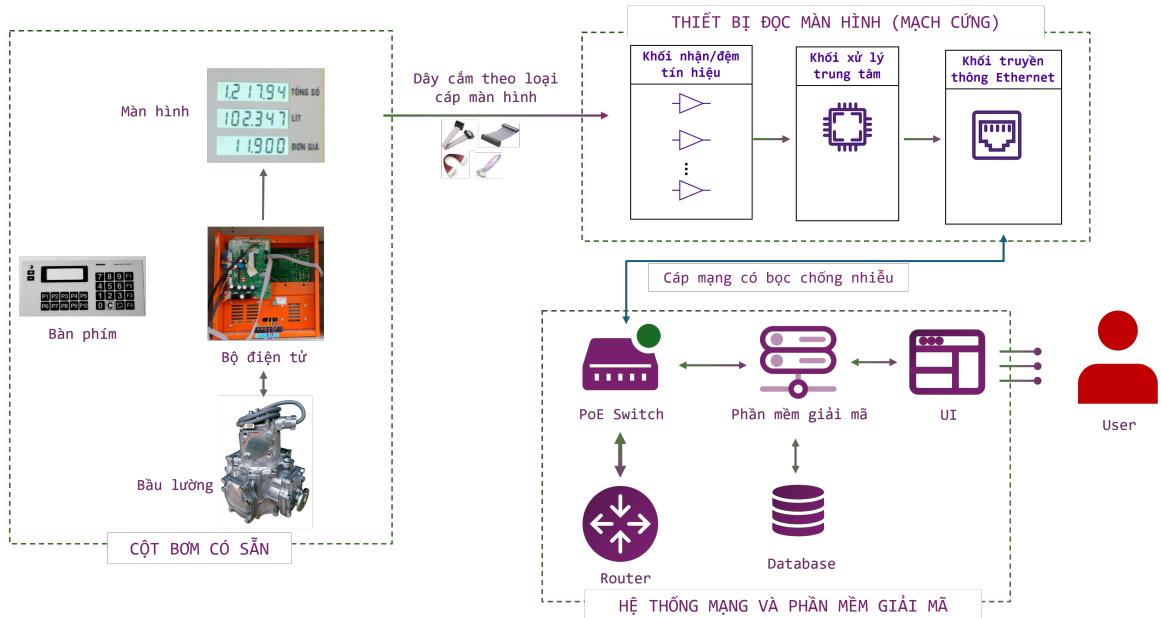
Đồng thời, chương này liệt kê các yêu cầu kỹ thuật cụ thể, cách tiếp cận bài toán từ đó đưa ra hướng thiết kế và triển khai tiếp theo.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1

CHƯƠNG 3. THIẾT KẾ HỆ THỐNG

3.1 Phân tích yêu cầu hệ thống



Hình 3.1. Các thành phần của hệ thống

Các thành phần chính của hệ thống bao gồm:

3.1.1 Thiết bị đọc ghi màn hình

Là mạch cứng gắn vào màn hình cột bơm, thu dữ liệu thô màn hình:

- Kích cỡ: vừa, có thể lắp đặt trong cột bơm
- Tần số lấy mẫu: 100Hz
- Cấp nguồn riêng
- Nguồn và tín hiệu vào được cách ly quang, đảm bảo không có tín hiệu quay ngược trở lại màn hình
- Có 2 relay đóng ngắt có thể nối tiếp với cột bơm
- Giao tiếp, gửi dữ liệu thông qua Ethernet
- (Optional) Đầu vào tín hiệu (cáp và bộ chuyển đổi tín hiệu) có thể thích hợp với nhiều loại màn hình khác ngoài ZCheng để phục vụ thu thập và phân tích thêm các loại cột bơm khác.

3.1.2 Phần mềm giải mã

Phần mềm giải mã đặt trong máy tính nội bộ tại trạm:

- Đọc và giải mã tín hiệu tới màn hình với tần số 100Hz.

- Giao diện phần mềm hiển thị màn hình đã giải mã theo thời gian thực
- Phát hiện, lưu trữ các phiên bơm. Giao diện hiển thị các phiên bơm đã lưu trữ.
- Điều khiển đóng ngắt relay.
- Có cơ chế cập nhật firmware (OTA) cho thiết bị.

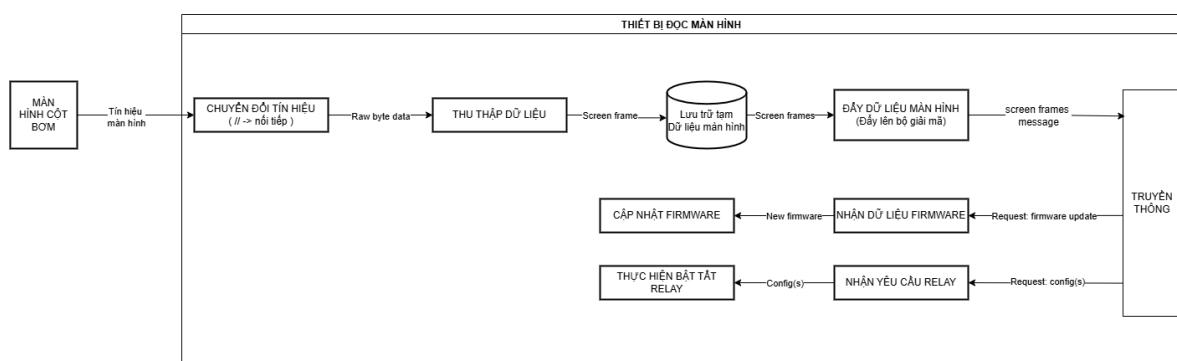
3.1.3 Hệ thống mạng

- POE Switch cấp nguồn riêng cho thiết bị đọc ghi màn hình
- Router điều khiển mạng LAN, cấp phát IP cho thiết bị
- Các thiết bị (thiết bị đọc ghi màn hình và phần mềm giải mã trong máy tính nội bộ) có thể tự động dò tìm và kết nối với nhau.

3.2 Mô hình thiết kế tổng thể

Phần này nêu ra sơ đồ khái niệm chức năng các thành phần hệ thống. Qua đó mô tả chức năng cụ thể của thành phần thông qua các khôi, đồng thời mô tả cách các khôi giao tiếp với nhau.

3.2.1 Sơ đồ khái niệm chức năng thiết bị đọc ghi màn hình (Device)



Hình 3.2. Sơ đồ khái niệm chức năng thiết bị đọc ghi màn hình (Device)

Giải thích các khôi chức năng:

• Chuyển đổi tín hiệu:

Nếu tín hiệu gửi tới màn hình cột bơm là nối tiếp, khôi này đệm tín hiệu nối tiếp tới bộ Thu thập dữ liệu

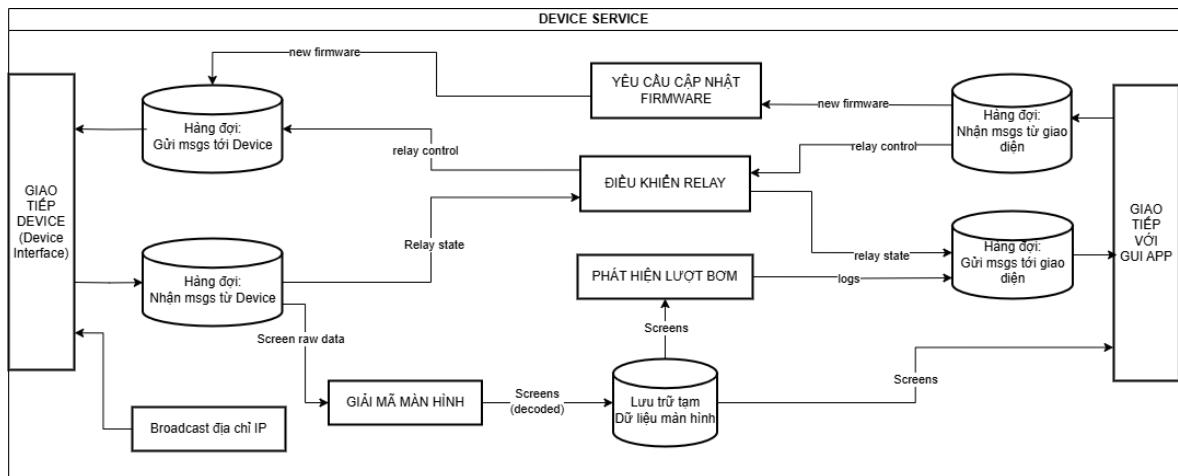
Nếu tín hiệu gửi tới màn hình cột bơm là song song, có nhiều chân dữ liệu, khôi này chuyển đổi tín hiệu từ các chân thành tín hiệu nối tiếp -> chuyển đổi thành các SPI frame để đệm tới bộ Thu thập dữ liệu.

• **Thu thập dữ liệu:** Thu thập các byte data tổng hợp thành các screen frame, lưu tạm vào 1 in-mem database.

• **Đẩy dữ liệu lên màn hình:** Thiết bị đọc màn hình tạo 1 message (request message) gồm nhiều screen frame để đẩy lên máy tính local

- **Nhận và cập nhật firmware mới:** Thiết bị đọc màn hình có thể nhận firmware mới từ máy tính local hoặc Logi Service và tự động cập nhật, khởi động lại
- **Nhận và cập nhật relay:** Thiết bị đọc ghi màn hình có thể nhận và thực hiện yêu cầu bật tắt relay
- **Truyền thông:** Lấy địa chỉ IP, dò tìm địa chỉ của phần mềm giải mã và tự động kết nối. Truyền nhận dữ liệu với phần mềm giải mã thông qua mạng LAN.

3.2.2 Sơ đồ khái niệm phần mềm giải mã (Device Service)



Hình 3.3. Sơ đồ khái niệm phần mềm giải mã và chốt phiên bơm (Device Service)

Giải thích các khái niệm chức năng:

- **Giao tiếp với Device (Thiết bị đọc ghi màn hình):**

Khối này được thiết kế chạy trên 1 luồng độc lập, nghe các yêu cầu kết nối từ thiết bị đọc ghi màn hình, kiểm tra và giữ kết nối nếu thiết bị hợp lệ

Truyền nhận gói tin với thiết bị đọc ghi và đẩy dữ liệu tới các khối khác để xử lý logic chính.

- **Giao tiếp với giao diện (GUI App):** Giữ kết nối và truyền nhận dữ liệu với phần mềm giao diện bao gồm:

Stream dữ liệu màn hình đã giải mã theo thời gian thực.

Gửi các phiên bơm (log) phát hiện được và trạng thái relay.

Nhận các yêu cầu từ giao diện: cập nhật firmware, relay.

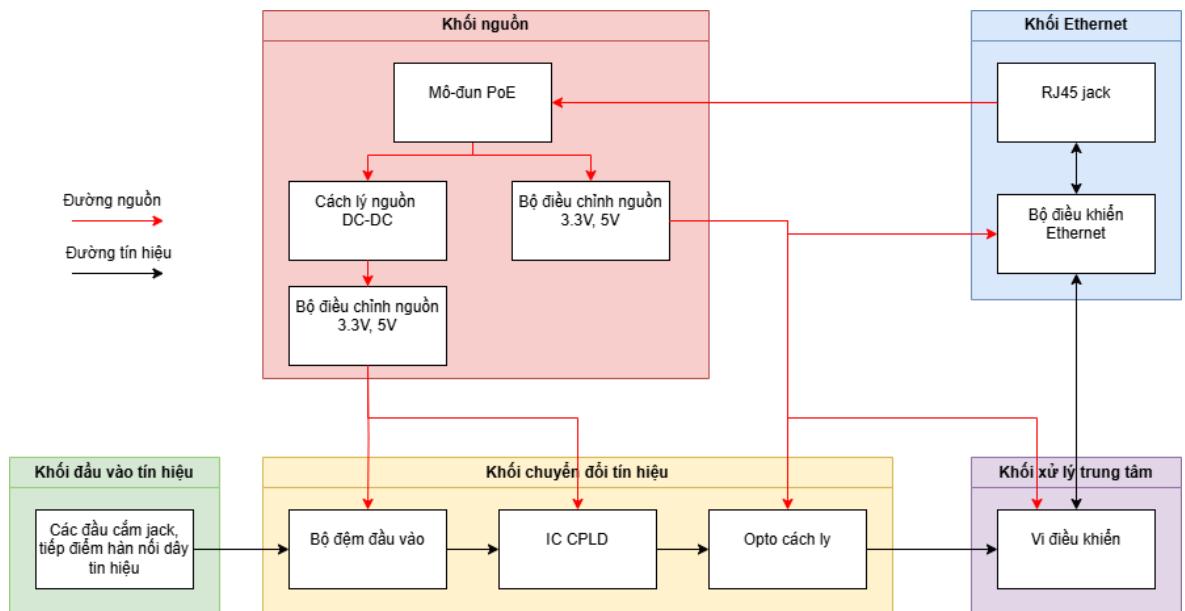
- **Giải mã màn hình và phát hiện phiên bơm:** Giải mã các gói tin dạng byte theo từng loại màn hình để được dữ liệu màn hình thời gian thực (số tiền, lít, đơn giá). Từ đó phát hiện trạng thái của máy (đang dừng và dừng trong bao nhiêu giây, đang tăng, reset) và phát hiện phiên bơm.

- **Cập nhật firmware:** Nhận file firmware từ giao diện, kiểm tra và thực hiện OTA với thiết bị đọc màn hình

- **Điều khiển relay:** Chuyển tiếp gói tin yêu cầu bật tắt relay tới thiết bị đọc ghi màn hình

3.3 Thiết kế phần cứng thiết bị đọc màn hình

3.3.1 Sơ đồ khái niệm phần cứng thiết bị đọc màn hình



Hình 3.4. Sơ đồ khái niệm phần cứng thiết bị đọc ghi màn hình

Thiết bị mạch cứng được cấp nguồn riêng bởi POE Switch. Nguồn và tín hiệu vào được các ly bởi bộ cách ly nguồn DC-DC, các bộ đếm đầu vào và Opto cách ly, đảm bảo không ảnh hưởng tới nguồn cấp của cột bơm cũng như không có tín hiệu hay nhiễu đẩy ngược về cột bơm.

Khối đầu vào tín hiệu và khối chuyển đổi tín hiệu cũng được thiết kế thêm để thu và khảo sát được nhiều loại màn hình khác, phục vụ cho việc mở rộng chức năng hệ thống.

Thiết bị giao tiếp với phần mềm giải mã thông qua mạng LAN, sử dụng khối Ethernet.

3.3.2 Lựa chọn linh kiện và vẽ sơ đồ nguyên lý

a. Khối xử lý trung tâm

Yêu cầu lựa chọn:

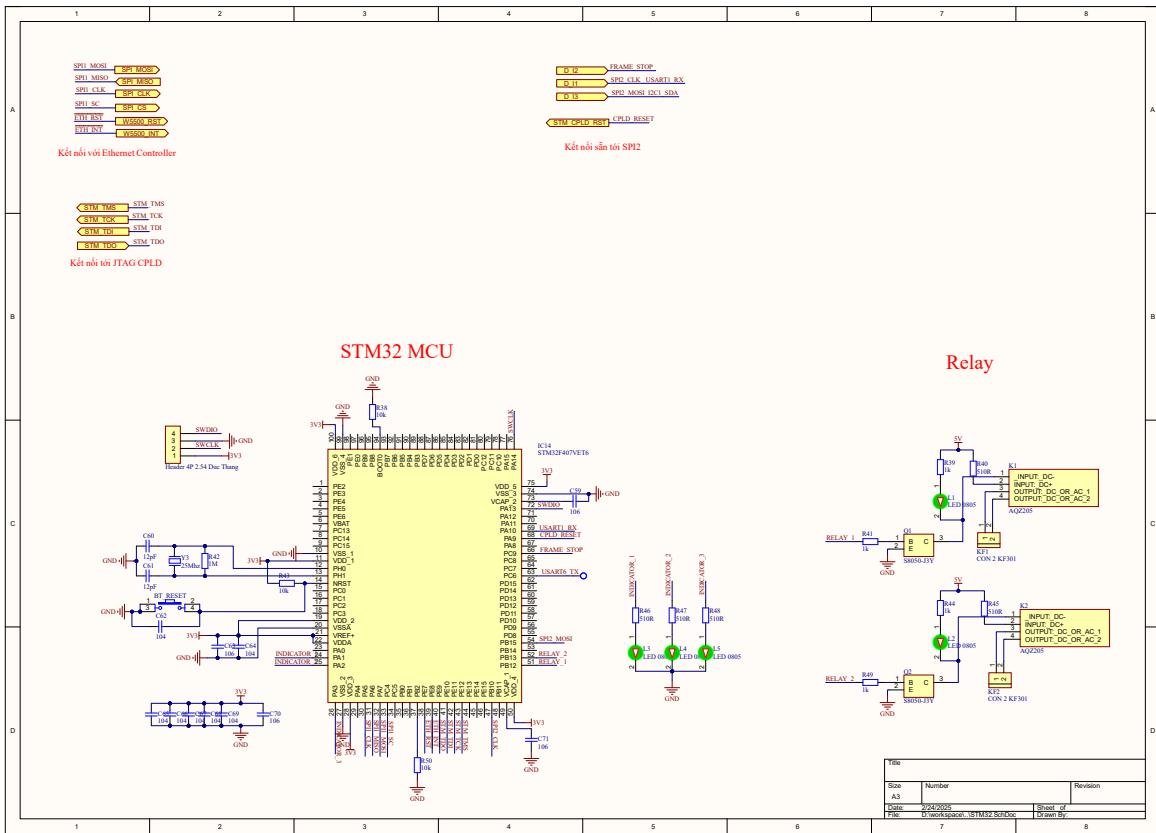
- Có 2 cổng giao tiếp SPI: mục đích để giao tiếp với khối chuyển đổi tín hiệu và với khối Ethernet.
- Bộ nhớ flash lớn hơn 220KB (chứa được 2 firmware, mỗi firm 110kB)

Từ những yêu cầu trên, lựa chọn STM32F407VET6 là vi điều khiển xử lý trung tâm. Các tính năng nổi bật của STM32F407VET6:

- Vi điều khiển sử dụng lõi ARM Cortex-M4 32-bit, tốc độ tối đa 168 MHz, tích hợp bộ xử lý dấu chấm động (FPU) và tập lệnh DSP, phù hợp cho các ứng dụng điều khiển và xử lý tín hiệu thời gian thực.
- Bộ nhớ Flash Main Memory 512 KB, được chia thành các sector như sau:
 - Sector 0-3: mỗi sector 16 KB
 - Sector 4: 64 KB
 - Sector 5, 6 và 11: mỗi sector 128-KB
- Cho phép linh hoạt khi lưu nhiều firmware (ví dụ dual-bank firmware upgrade), hoặc phân vùng cho bootloader, data log, cấu hình...
- Tích hợp 3 cổng SPI (SPI1, SPI2, SPI3), hỗ trợ chế độ master/slave, tốc độ tối đa đến 42 MHz (SPI1) và 21 MHz (SPI2/SPI3). Phù hợp cho các giao tiếp tốc độ cao như ADC ngoại vi và module Ethernet SPI.
- Có tổng cộng 17 bộ Timer, gồm 12 timer 16-bit và 2 timer 32-bit. Hỗ trợ nhiều chế độ: PWM, encoder, input capture, output compare, rất hữu ích trong điều khiển động cơ, đo thời gian hoặc phát xung.
- Hỗ trợ nhiều chuẩn giao tiếp ngoại vi: UART, I2C, USB OTG, CAN, SDIO, Ethernet MAC - thuận tiện cho mở rộng và kết nối các module chức năng khác.
- Hoạt động với điện áp 1.8 V đến 3.6 V, hỗ trợ các chế độ tiết kiệm năng lượng như Sleep, Stop, và Standby - phù hợp cho ứng dụng nhúng có yêu cầu tiêu thụ điện thấp.

Sơ đồ nguyên lý mạch:

- Nguồn cấp 3.3V được đưa vào các chân VDD và Vref của module, các chân VSS được nối đất. Thiết kế nguồn nối với đất qua các tụ Bypass có nhiệm vụ lọc nhiễu cao tần cho nguồn nuôi vi điều khiển.
- Sử dụng SPI1 để giao tiếp với chip Ethernet. SPI2 để giao tiếp với khối chuyển đổi dữ liệu
- Các chân PA1, PA2, PA3 điều khiển đèn báo led
- Reset STM32 được thực hiện khi chân NRST hạ xuống mức 0:
 - Điện trở kéo lên 10KΩ
 - Tụ điện: C = 100nF, giúp tạo xung reset ngắn nhưng đủ để tái khởi STM32
- Mạch nạp cho STM32:
 - Thiết kế mạch nạp theo chuẩn SWD



Hình 3.5. Sơ đồ nguyên lý khối xử lý trung tâm

– Hỗ trợ nạp thông qua mạch nạp STLink

Khối đầu vào tín hiệu

Yêu cầu lựa chọn:

Ngoài việc thu và giải mã loại màn hình ZCheng, khối đầu vào tín hiệu còn cần hỗ trợ các loại đầu vào cho màn hình khác để thu mẫu dữ liệu, phục vụ cho việc giải mã, mở rộng hệ thống sau này. Các loại đầu vào màn hình đã khảo sát được bao gồm:

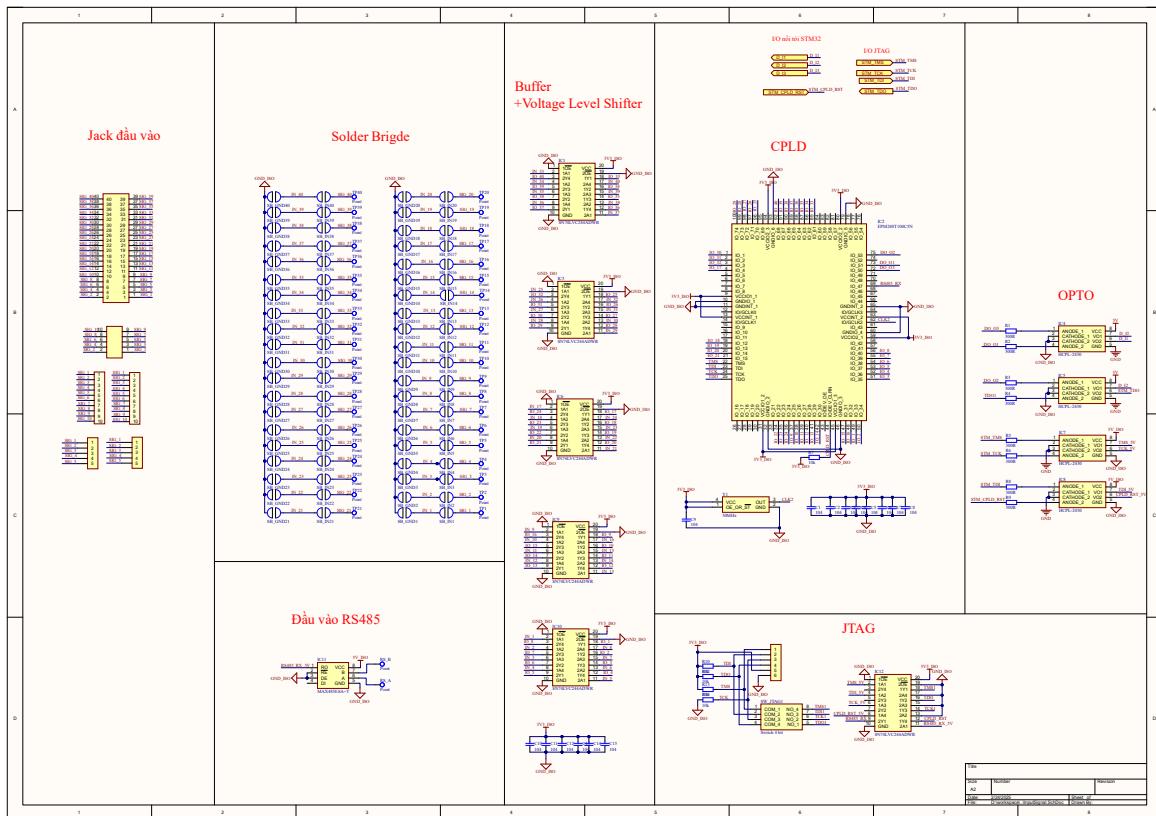
- Đầu vào SPI (cho màn hình ZCheng, đã giải mã được). Loại cáp: 8P 3.96mm. Số chân tín hiệu: 3
- Loại cáp: 5P 3.96mm, số chân tín hiệu: 2
- Loại cáp: IDE 10. Số chân tín hiệu 3
- Loại cáp: IDE 40. Số chân tín hiệu 30

Với danh sách loại đầu vào ở trên, thiết kế khối đầu vào tín hiệu gồm các đầu jack cắm và tiếp điểm hàn để đấu nối dây điện. Khối gồm 2 phần:

- Các đầu cắm cáp: hỗ trợ các loại cáp đã khảo sát.

- Các điểm hàn nối dây: Trường hợp các loại màn hình khác, sẽ hàn các đường tín hiệu đó thông qua các đầu tiếp điểm hàn, kèm dây điện nối (nếu cần). Vị trí mỗi hàn, đầu dây sẽ cố định theo từng loại tín hiệu màn hình.

Tổng số chân cho khối đầu vào tín hiệu là 40 chân.



Hình 3.6. Sơ đồ nguyên lý khối đầu vào tín hiệu

Khối nguồn và Ethernet

Yêu cầu thiết kế:

- Khối nguồn:

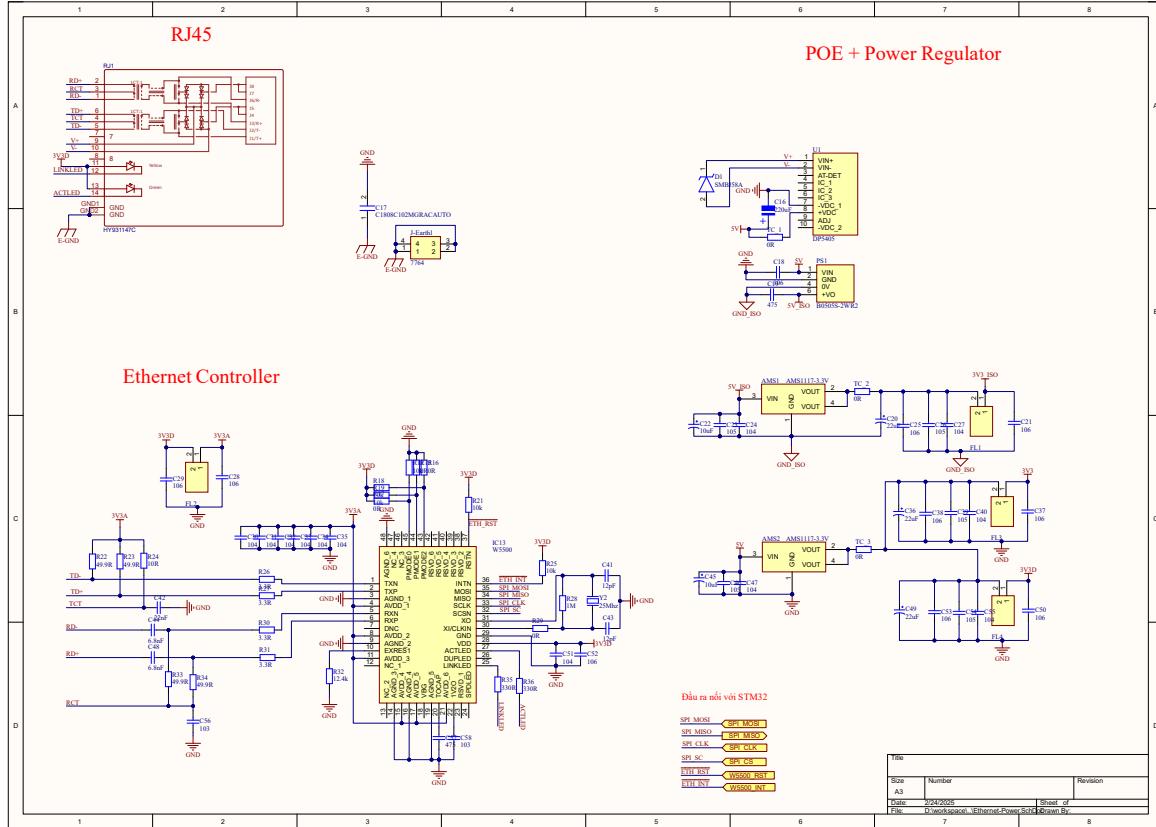
- Cấp nguồn 3.3V cho khối khác. Chuyển đổi nguồn 5V của POE sang 3.3V.
- Cách ly và chống nhiễu.

- Khối Ethernet:

- Chống nhiễu.
- Giao tiếp với MCU (STM32) theo chuẩn SPI.

Với những yêu cầu trên, lựa chọn:

- B0505S-2WR2 và AMS1117-3.3V để cách ly và chuyển đổi nguồn POE thành nguồn 3.3V.
- Chip W5500 để giao tiếp Ethernet.



Hình 3.7. Sơ đồ nguyên lý khối nguồn và giao tiếp Ethernet

Giải thích khối nguồn:

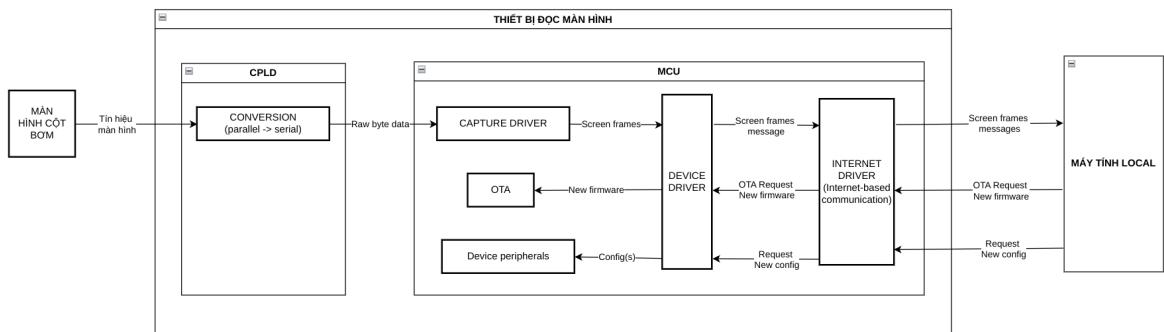
- Mô-đun DP5405 để nhận nguồn từ PoE Switch (qua jack RJ45), chuyển đổi sang điện áp 5VDC để cấp nguồn cho toàn bộ thiết bị.
- IC B0505S: Có chức năng chuyển đổi điện áp 5VDC sang điện áp 5VDC cách ly. Việc cách ly nguồn để góp phần cách ly tín hiệu điện giữa tín hiệu của cáp màn hình với tín hiệu khác trong mạch, để 2 loại tín hiệu này không ảnh hưởng tới nhau.
- IC AMS1117-3.3V: Là IC điều chỉnh nguồn, đầu ra IC sẽ cung cấp nguồn điện áp 3.3V cho các linh kiện khác trong mạch. Có 2 IC trong đó IC thứ nhất để chuyển đổi điện áp 5VDC sang 3.3VDC, IC thứ hai chuyển đổi điện áp 5VDC cách ly sang 3.3V, từ đó điện áp 3.3V này cũng được cách ly với điện áp 3.3VDC ở IC thứ nhất.

Giải thích khối Ethernet:

- Jack cắm Ethernet: Sử dụng jack RJ45 HY931147C để cắm dây Ethernet vào thiết bị, vừa để dẫn nguồn cho thiết bị, vừa để hỗ trợ giao tiếp mạng Ethernet.
- IC điều khiển Ethernet: Sử dụng IC W5500 để cung cấp các chức năng liên quan đến truyền/nhận dữ liệu thông qua Ethernet cho khối Xử lý trung tâm.

3.4 Thiết kế firmware thiết bị đọc màn hình

3.4.1 Kiến trúc triển khai firmware



Hình 3.8. Sơ đồ khái niệm khai firmware cho thiết bị

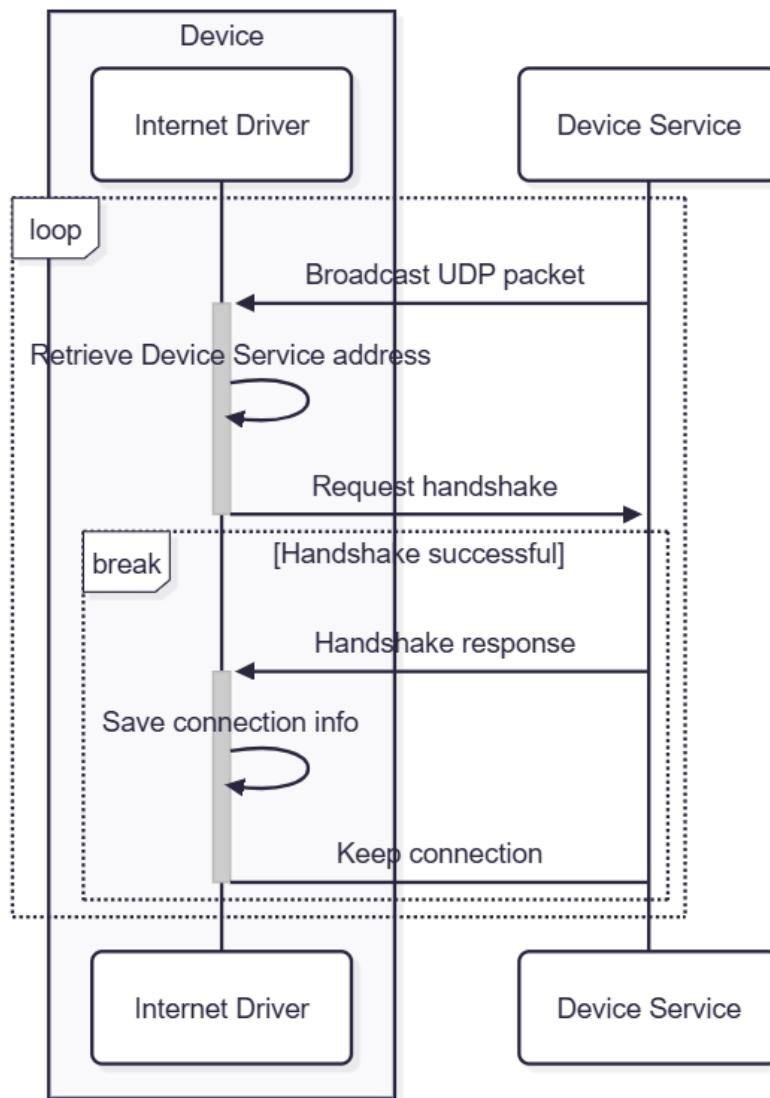
Các khái niệm khai thực hiện các chức năng đã được nêu ra trong [hình 3.2](#).

Mỗi khái niệm khai ở [hình 3.8](#). được tổ chức thành một module mã nguồn, xử lý và trao đổi dữ liệu với nhau.

- Capture Driver: Nhận các SPI frame từ CPLD, đóng gói thành các dữ liệu màn hình dạng byte data, cùng thời gian nhận dữ liệu và loại màn hình.
- Device Driver: Xử lý giao thức. Xử lý các gói tin nhận được (dạng byte frame) từ đầu vào Ethernet và chuyển tiếp tới các khái niệm Capture Driver, OTA và Device peripherals để thực hiện các logic chính. Đồng thời đẩy các gói tin màn hình (dạng byte frame) cho khái niệm Internet để thực hiện gửi cho phần mềm máy tính (Device Service)
- Internet Driver: Giao tiếp với chip W5500 để thực hiện trao đổi gói tin qua mạng LAN.
 - Xin cấp phát IP tĩnh từ router
 - Nhận địa chỉ IP của phần mềm máy tính (Device Service) thông qua gói tin được broadcast
 - Trao đổi dữ liệu dạng byte frame với phần mềm máy tính (Device Service) thông qua giao thức TCP, cấu trúc gói tin tuân theo API đã được quy ước.
 - Khi bị mất kết nối với phần mềm Device Service, cố gắng kết nối trở lại với phần mềm.

3.4.2 Trình tự giao tiếp giữa các khối

a. Nhận gói tin broadcast và kết nối tới phần mềm Device Service



Hình 3.9. Trình tự giao tiếp: Thiết lập kết nối tới phần mềm giải mã Device Service

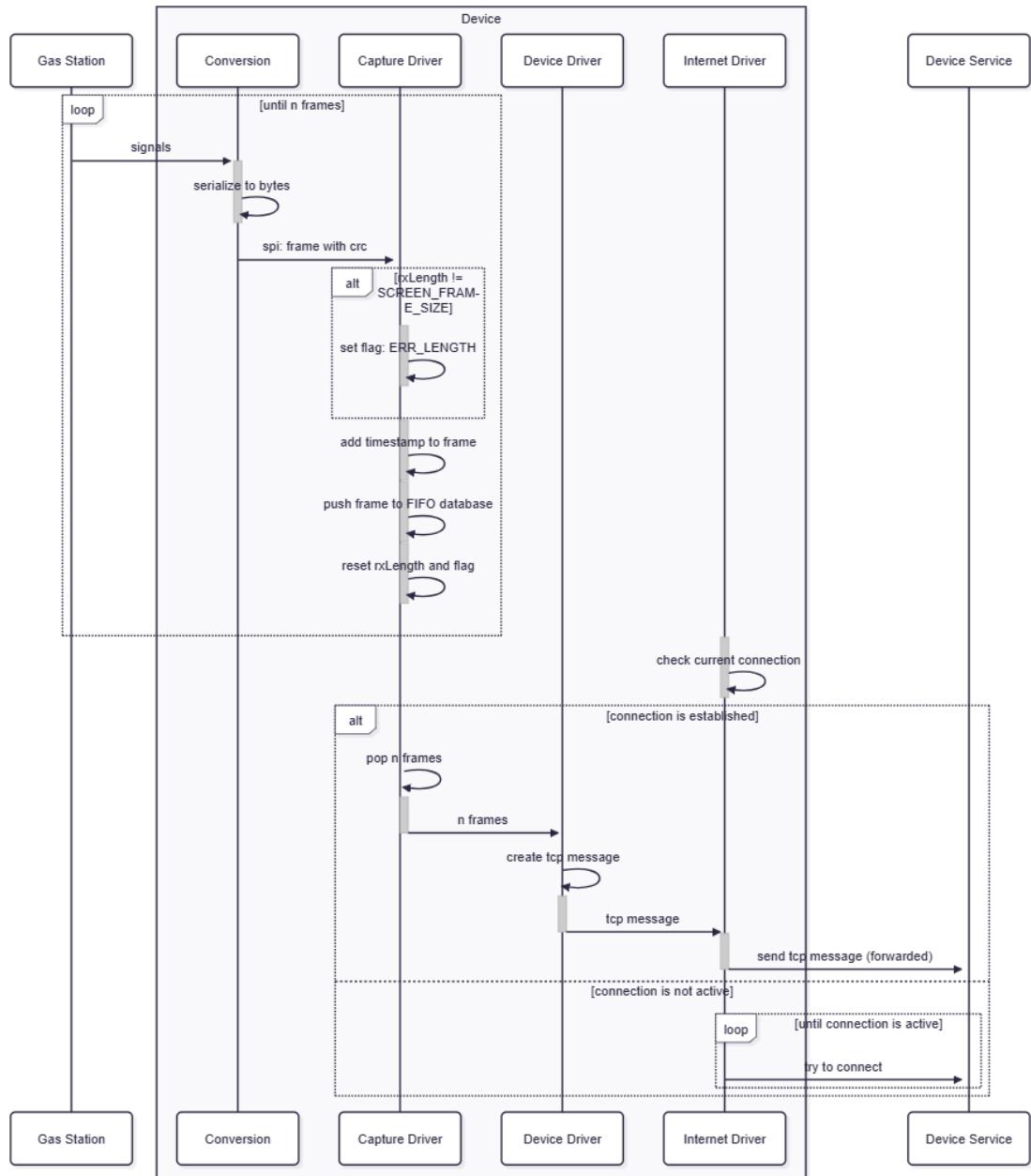
Gói tin broadcast của phần mềm giải mã (Device Service) có chứa địa chỉ (IP và PORT) của phần mềm. Thiết bị nhận được gói tin, lấy thông tin địa chỉ và gửi yêu cầu thiết lập kết nối tới phần mềm (handshake).

Trong trường hợp không kết nối được hoặc mất kết nối, thiết bị sẽ liên tục yêu cầu kết nối trở lại tới phần mềm.

b. Đẩy dữ liệu thô màn hình (dạng byte)

Tín hiệu màn hình được thu thập bởi bộ chuyển đổi tín hiệu, và đẩy tới **Capture Driver**, sử dụng SPI.

Khối Capture Driver nhận từng byte, lưu trữ tạm vào 1 buffer cho đến khi độ dài



Hình 3.10. Trình tự giao tiếp: thu thập và đẩy dữ liệu màn hình

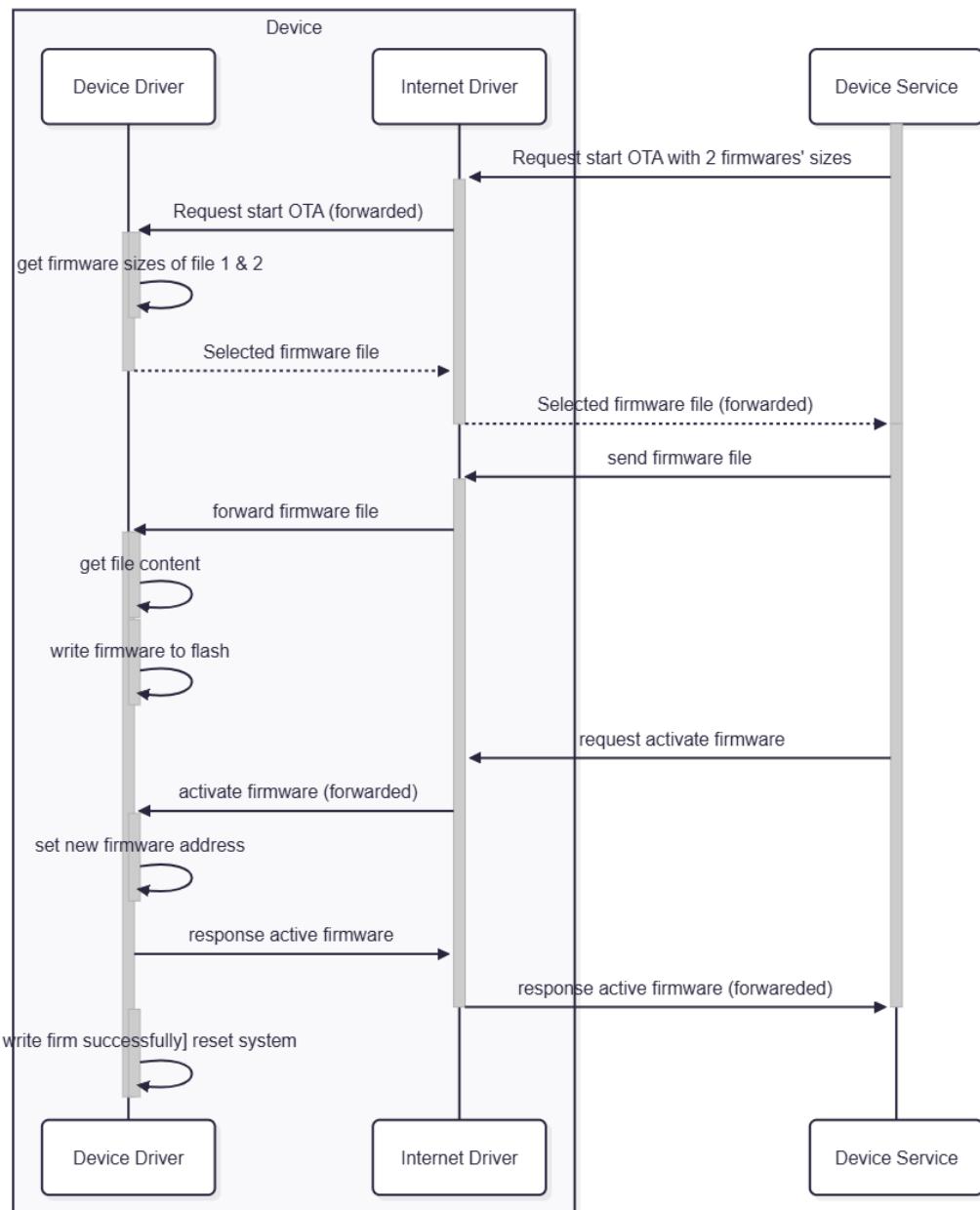
buffer băng độ dài tiêu chuẩn 1 frame màn hình (22 byte đối với máy ZCheng). Khi đó, dữ liệu buffer được lấy ra để tạo thành 1 frame màn hình.

Mỗi frame màn hình chứa dữ liệu "tổng tiền", "số lít", "đơn giá" đã được mã hóa, sẽ được lưu tạm vào 1 FIFO database (tại memory) cùng với thời gian nhận frame. Buffer nhận sau đó được xóa để bắt đầu nhận frame mới.

Do tần số nhận frame cao (100Hz), không thể gửi dữ liệu với tần số như vậy thông qua chip Ethernet, dữ liệu màn hình được lưu tạm vào 1 FIFO Database. Ở mỗi

lần gửi, **Internet Driver** kiểm tra kết nối hiện tại với phần mềm Device Service. Nếu kết nối vẫn tốt, **Capture Driver** sẽ lấy nhiều frame từ database cùng một lúc để gửi đến phần mềm Device Service.

c. Cập nhật firmware



Hình 3.11. Trình tự giao tiếp: cập nhật firmware từ xa (OTA)

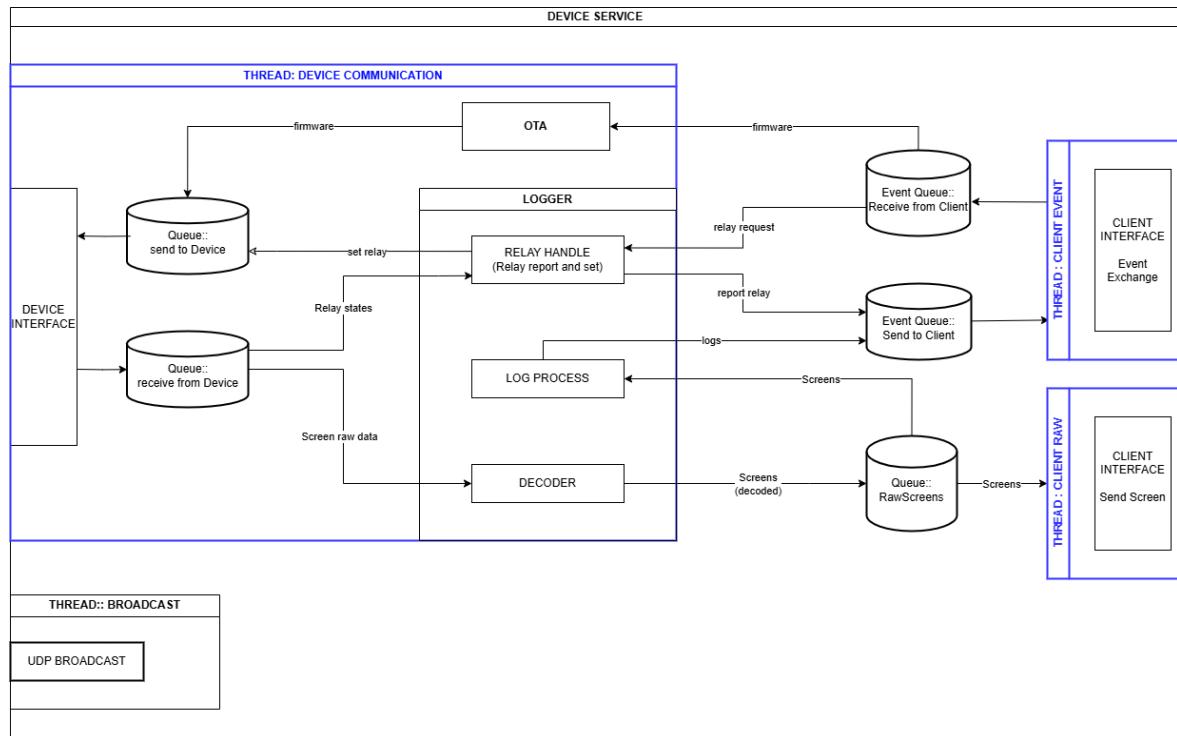
Phần mềm Device Service gửi yêu cầu bắt đầu tiến trình OTA cùng với kích thước firmware mới. Thiết bị thông báo lại tên file firmware muốn cập nhật (file 1 hoặc 2).

Sau đó, Device Service gửi từng đoạn dữ liệu firmware mới cùng offset. Thiết bị dữ liệu vào flash, tại vị trí cho firmware mới.

Khi việc truyền nhận firmware mới hoàn tất, thiết bị đặt con trỏ chứa địa chỉ firmware theo giá trị địa chỉ mới và tiến hành reset.

3.5 Thiết kế phần mềm giải mã Device Service

3.5.1 Kiến trúc triển khai phần mềm



Hình 3.12. Sơ đồ khái niệm khai triển phần mềm Device Service

Các khái niệm khai triển thực hiện các chức năng đã nêu ở sơ đồ khái niệm (hình 3.3.). Về mặt triển khai, Device Service là phần mềm đa luồng, chạy dưới dạng tiến trình chạy ngầm trên máy tính nội bộ, trong đó có các luồng chính:

- Device Communication giữ kết nối và giao tiếp và điều khiển thiết bị đọc ghi màn hình (Device), thực hiện giải mã và chốt phiên bơm.
- Client Event: luồng này trao đổi các sự kiện với phần mềm giao diện bao gồm:
 - Các phiên bơm (gọi là log).
 - Yêu cầu bật tắt relay (từ phía giao diện) và thông báo trạng thái relay (từ phía thiết bị) cho giao diện.
 - Yêu cầu cập nhật firmware và nội dung file firmware.
- Client Raw: luồng riêng biệt để gửi trực tiếp các màn hình đã giải mã theo thời gian thực, để giao diện có thể hiển thị giá trị tương ứng với màn hình cây xăng thực tế.

- Broadcast: luồng đọc lập để broadcast gói tin UDP chứa địa chỉ (IP và PORT) của Device Service này cho toàn bộ thiết bị trong mạng LAN. Các thiết bị đọc ghi màn hình nhận được địa chỉ và chủ động kết nối.

Các luồng giao tiếp với nhau thông qua các hàng đợi (Queue). Các hàng đợi này được thiết kế để thread-safe, tức là các luồng chạy song song có thể truy cập vào hàng đợi mà không xảy ra xung đột.

3.5.2 Trình tự giao tiếp và thuật toán cho các khối

Phần này trình bày trình tự giao tiếp giữa các khối trong các use case cụ thể, đồng thời trình bày các lưu đồ thuật toán quan trọng. Các use case chính bao gồm:

- Giải mã và gửi giá trị màn hình theo thời gian thực
- Cập nhật firmware cho thiết bị
- Gửi yêu cầu đóng ngắt relay

a. Giải mã, gửi giá trị màn hình và phát hiện phiên bơm

Việc giải mã tiến hành qua các bước sau:

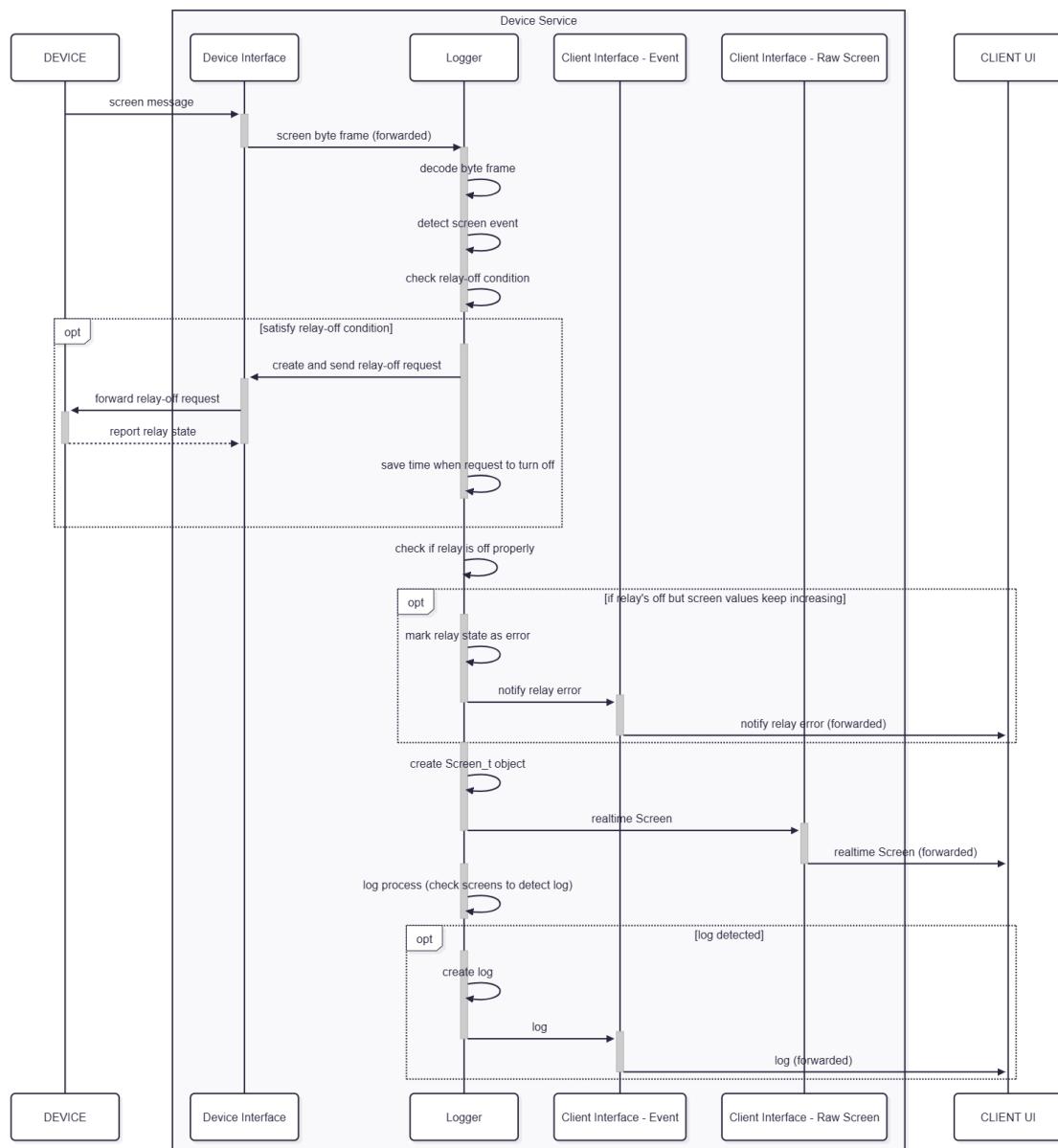
- Giải mã dữ liệu thô dạng byte thành giá trị màn hình (số tiền, số lít, đơn giá).
- Phân loại sự kiện màn hình (screen event):
 - Phát hiện trạng thái máy (machine state).
 - Phát hiện phiên bơm (log) nếu có.

Dữ liệu thô từ màn hình cây xăng được đọc liên tục, decode và so sánh để xác định giá trị và độ biến thiên giá trị đang hiển thị ở cột bơm ('bằng 0 | đang tăng | đang dừng'), xác định trạng thái hiện tại của cột bơm ('reset | đang bơm | tạm dừng bơm'). Từ đó đưa ra quyết định chốt một lượt bơm - gọi là 1 'log'.

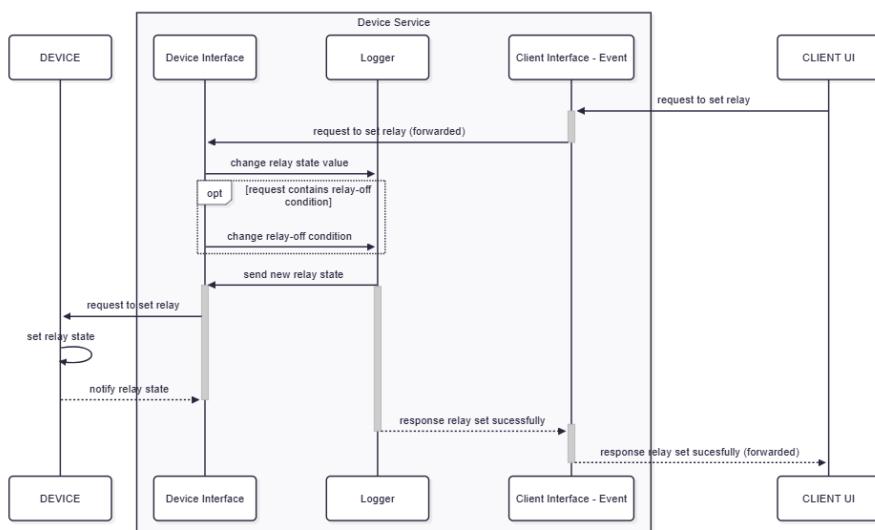
b. Điều khiển relay cho thiết bị

Tùy giao diện người dùng Client UI, có thể:

- Gửi yêu cầu bật tắt relay.
- Đặt ngưỡng giới hạn trên (số tiền hoặc số lít) để relay tự động tắt.



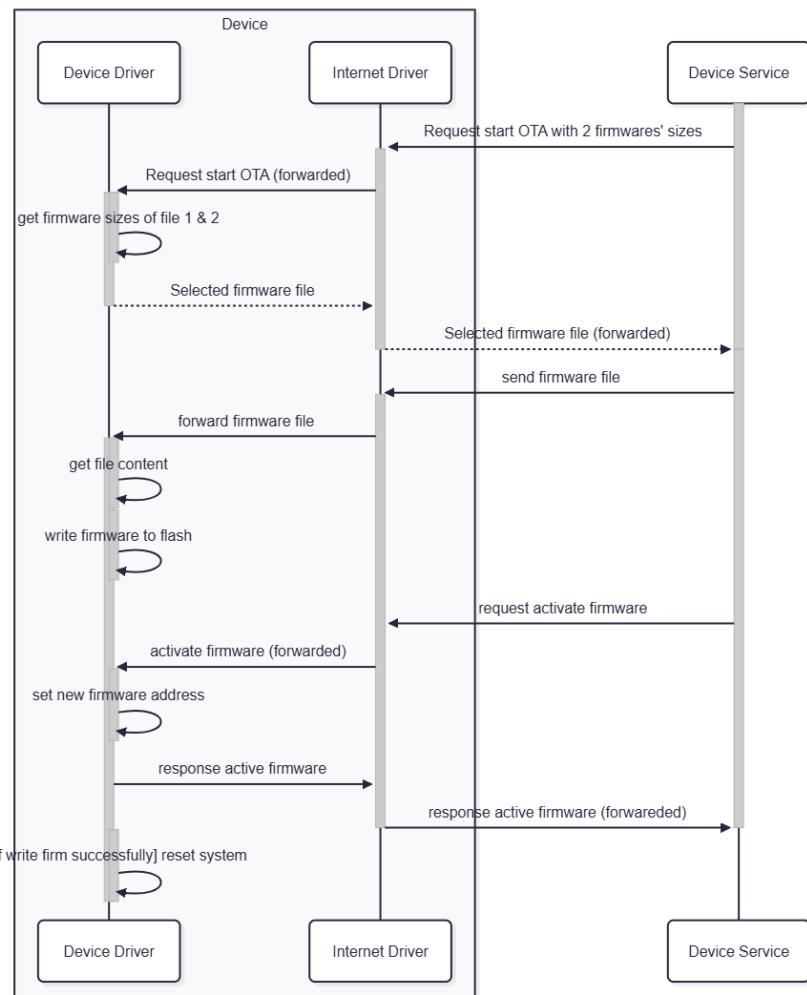
Hình 3.13. Trình tự giao tiếp cho phần mềm giải mã: Giải mã màn hình



Hình 3.14. Trình tự giao tiếp phần mềm trong trường hợp điều khiển relay

c. Cập nhật firmware từ xa (OTA)

File firmware mới được gửi từ Client UI, được mã hóa dưới dạng base64 string. Phần mềm Device Service giải mã và kiểm tra hợp lệ (verify hash) rồi lưu tạm vào RAM. Sau khi nhận được yêu cầu kích hoạt quá trình OTA, Device Service bắt đầu gửi từng đoạn firmware mới tới thiết bị đọc ghi màn hình.



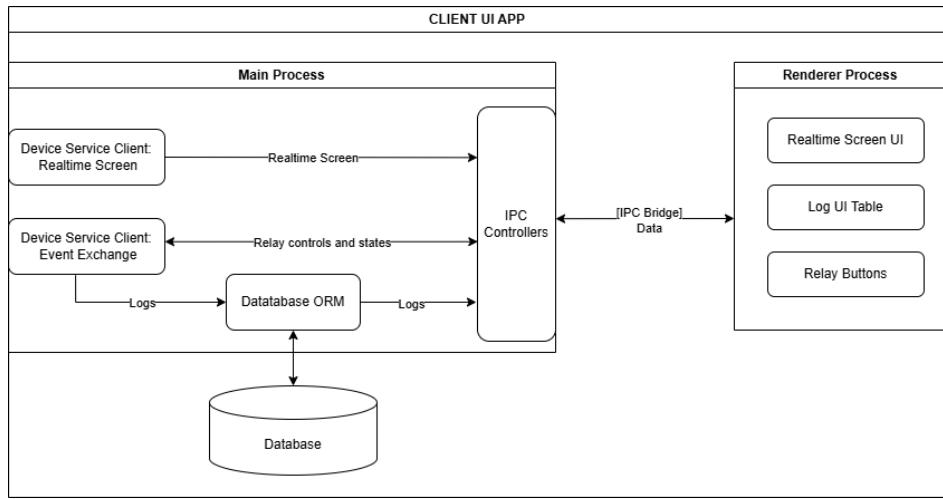
Hình 3.15. Trình tự giao tiếp phần mềm trong trường hợp thực hiện OTA

3.6 Giao diện điều khiển cho người dùng

Phần mềm giao diện được thiết kế cho người dùng cuối để lưu trữ, hiển thị và điều khiển phần mềm giải mã Device Service.

Yêu cầu thiết kế:

- Giao diện trực quan, gần gũi với người dùng.
- Độc lập với Device Service, kiến trúc có thể dễ dàng mở rộng, hướng tới trở thành công cụ trực quan có thể quan sát và hỗ trợ debug cho Device Service.
- Tương tác với cơ sở dữ liệu để đọc ghi các phiên bơm.

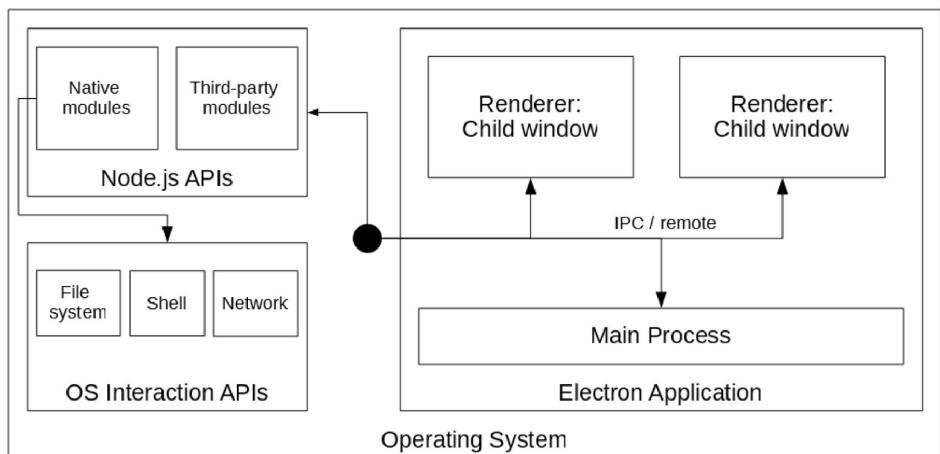


Hình 3.16. Sơ đồ khái niệm khai triển phần mềm giao diện

Phần mềm giao diện xây dựng trên môi trường Nodejs, sử dụng framework Electron để tạo một web-based Desktop App. Phần mềm gồm 2 tiến trình (process) chính:

- Main Process có thể tương tác trực tiếp với hệ điều hành thông qua môi trường NodeJS. Sử dụng để:
 - Mở các cổng TCP hứng dữ liệu (màn hình thời gian thực, các events) từ phần mềm giải mã Device Service.
 - Xử lý các logic, tương tác với cơ sở dữ liệu: đọc và ghi các phiên bơm (log).
 - Chuyển tiếp các dữ liệu lên phía Renderer Process để hiển thị lên giao diện
- Renderer Process là tiến trình sử dụng để hiển thị giao diện, tương tác với người dùng. Tiến trình này bản chất là 1 module có lõi là Chromium. Bản chất của việc render và hiển thị sử dụng các công nghệ web browser (HTML, CSS, JavaScript) để vẽ các element. Do đó, việc tương tác trực tiếp với hệ điều hành cũng bị hạn chế, cần có Main Process và IPC (InterProcess Communication) để người dùng thực hiện các tác vụ với hệ điều hành và cơ sở dữ liệu thông qua giao diện.

Phần mềm có một cơ chế IPC (InterProcess Communication) riêng để các process giao tiếp với nhau, giúp chia sẻ dữ liệu an toàn.



Hình 3.17. Giao tiếp giữa các process của ứng dụng thông qua IPC

3.7 Tổng kết chương

Trong chương này đã trình bày các khái niệm của các thành phần hệ thống. Từ yêu cầu về chức năng đó, đưa ra thiết kế chi tiết bao gồm:

- Sơ đồ nguyên lý mạch cứng
- Sơ đồ triển khai phần mềm
- Tương tác giữa các khái niệm trong phần mềm.

Và diễn giải các sơ đồ.

CHƯƠNG 4. TRIỂN KHAI VÀ THỬ NGHIỆM

4.1 Quy trình triển khai

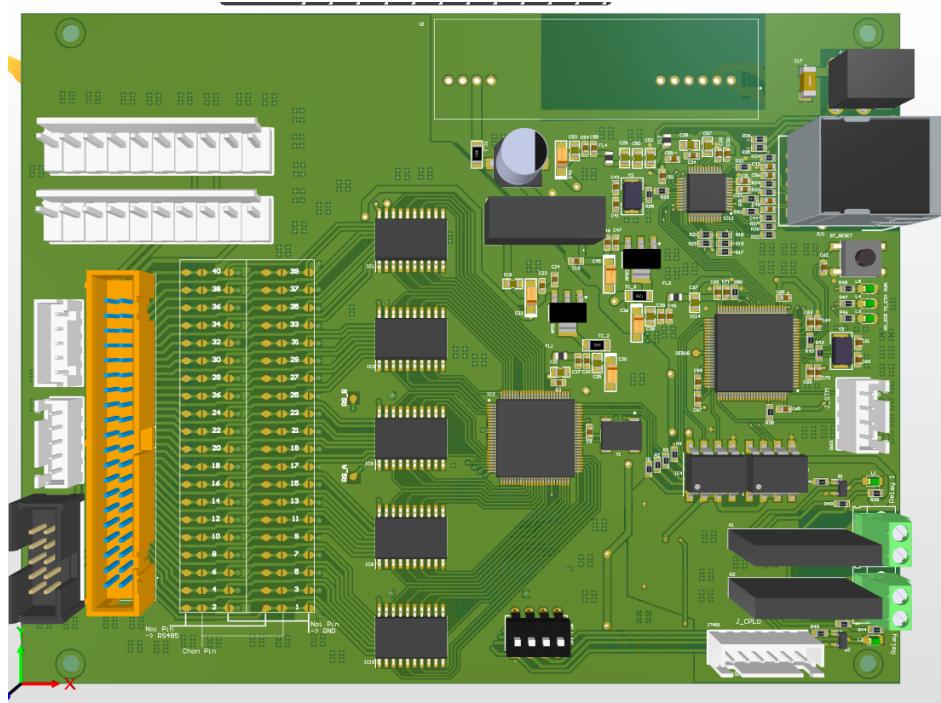
Ở chương này em trình bày chi tiết quá trình triển khai, xây dựng sản phẩm sử dụng các công nghệ đã lựa chọn ở Chương 2 và dựa trên thiết kế chi tiết đã nêu ra ở Chương 3.

Nội dung gồm quy trình triển khai, mô tả đầu ra sản phẩm, kiểm thử và chạy với dữ liệu mô phỏng, lắp ráp và kiểm thử tại hiện trường.

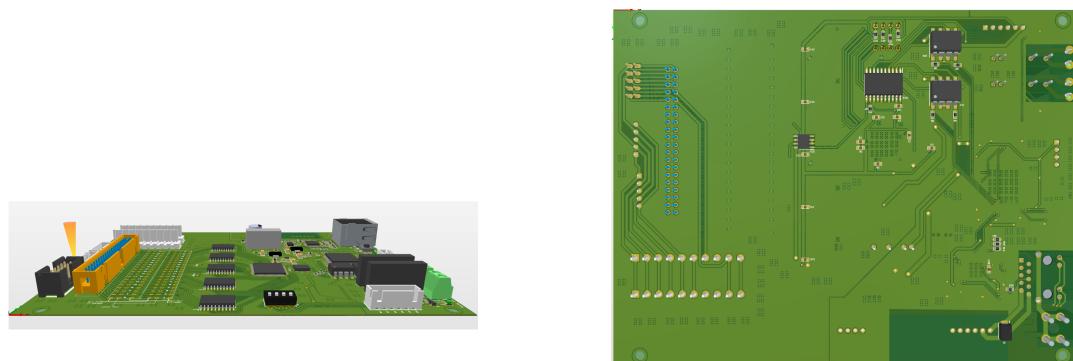
4.1.1 Mạch phần cứng thiết bị đọc màn hình

a. Vẽ mạch PCB từ sơ đồ nguyên lý

Từ sơ đồ nguyên lý đã nêu ở Chương 3, tiến hành vẽ mạch PCB trên Altium:

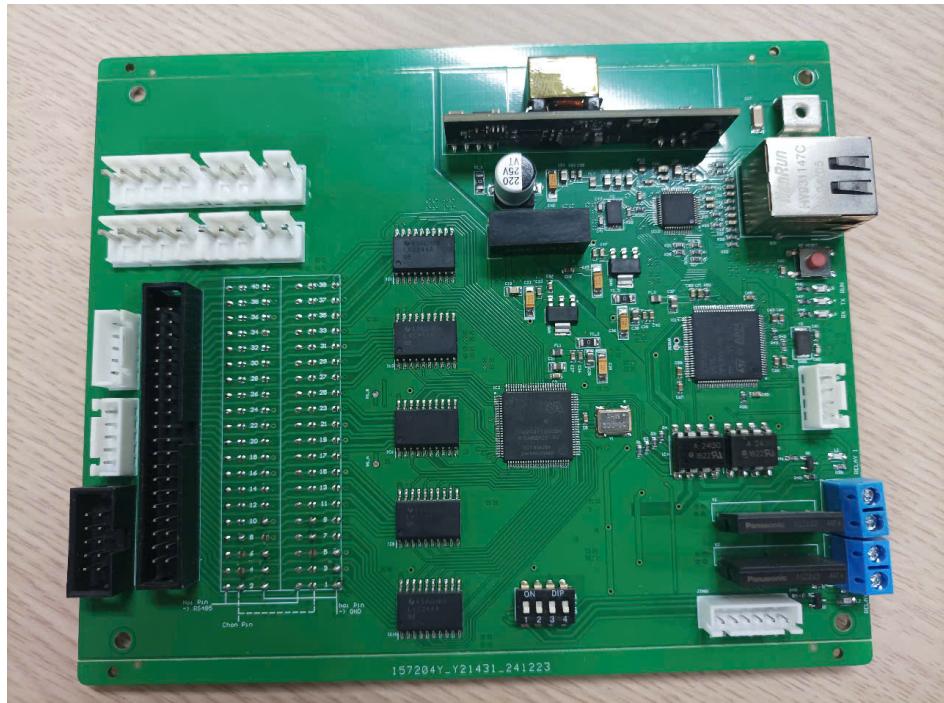


Hình 4.1. Mạch PCB - Mặt trên

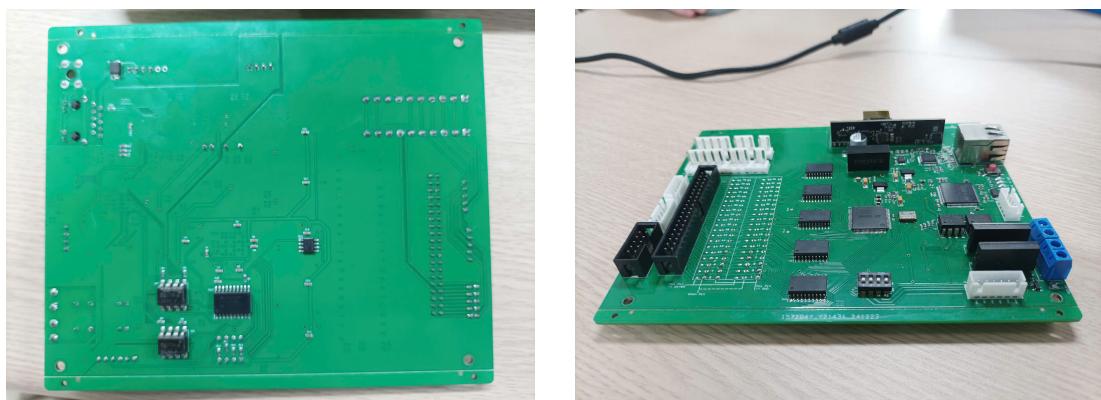


Hình 4.2. Mạch PCB - Mặt ngang và mặt dưới

b. In và hàn mạch thực tế



Hình 4.3. Sản phẩm thiết bị đọc ghi màn hình cột bơm: mặt trên



Hình 4.4. Sản phẩm thiết bị đọc ghi màn hình cột bơm: mặt ngang và dưới

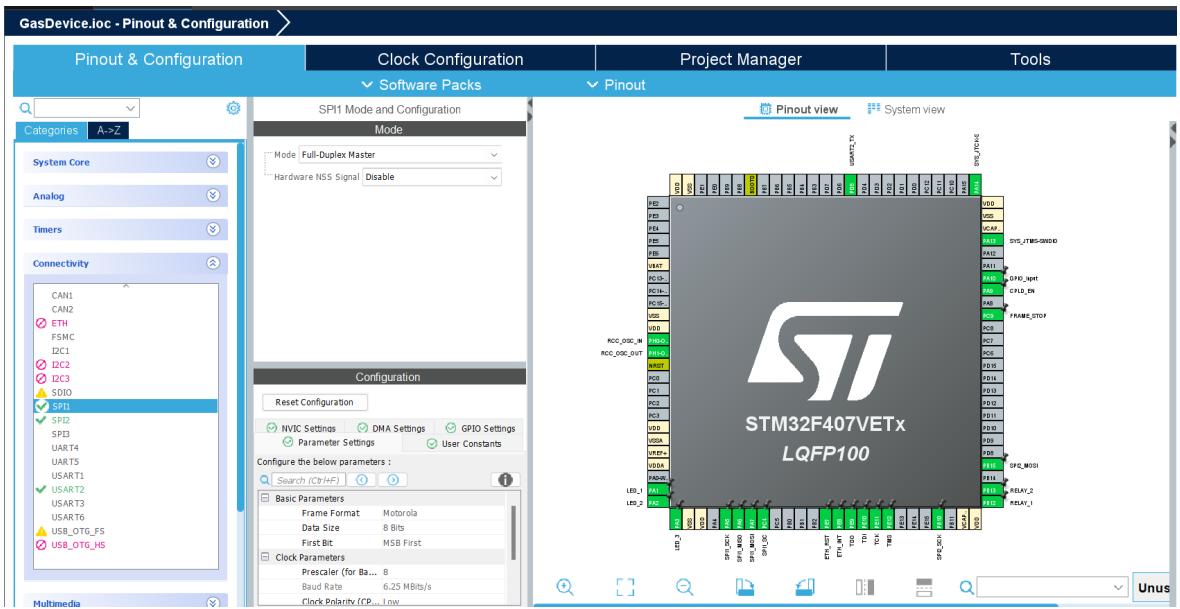
4.1.2 Firmware thiết bị đọc màn hình

Môi trường/ công cụ phát triển để lập trình cho STM32: STM32CubeIDE.

Setup các cổng IO

Các ngoại vi được sử dụng bao gồm:

- SPI1: Kết nối với module W5500, giao tiếp Ethernet.
- SPI2: Kết nối với CPLD, nhận dữ liệu màn hình dạng byte data.
- USART2: Sử dụng để debug (kết nối với máy tính và gửi thông báo lên màn hình terminal)
- Các cổng IO khác:



Hình 4.5. Triển khai: cấu hình các chân trong STM32CubeIDE

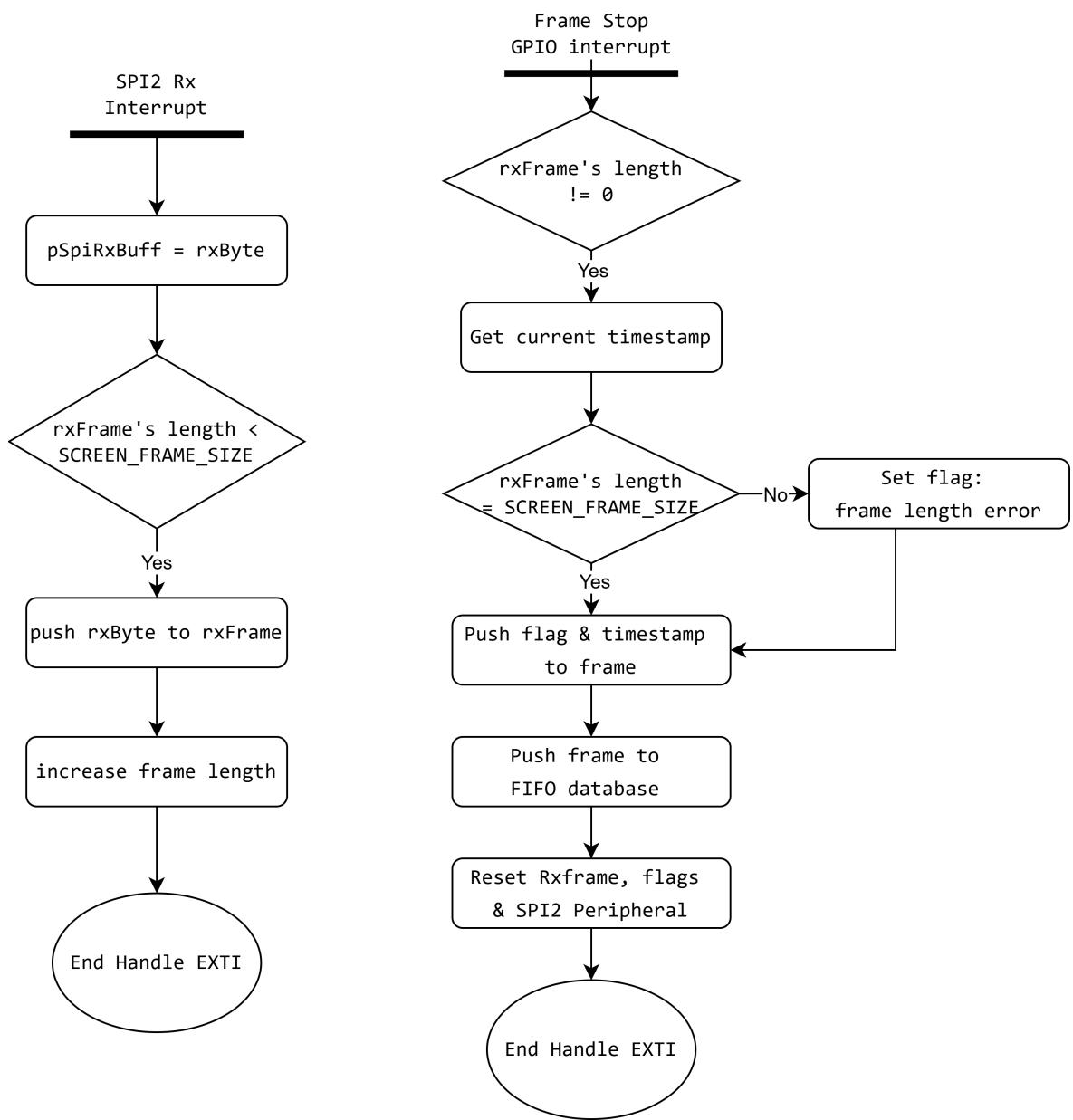
- PB12 và PB13: điều khiển relay 1 và 2.
- PA1, PA2 và PA3: các đèn báo led.
- PA9: CPLD Enable
- PC9: Frame Stop - Ngắt GPIO đầu vào báo kết thúc 1 frame

Triển khai firmware

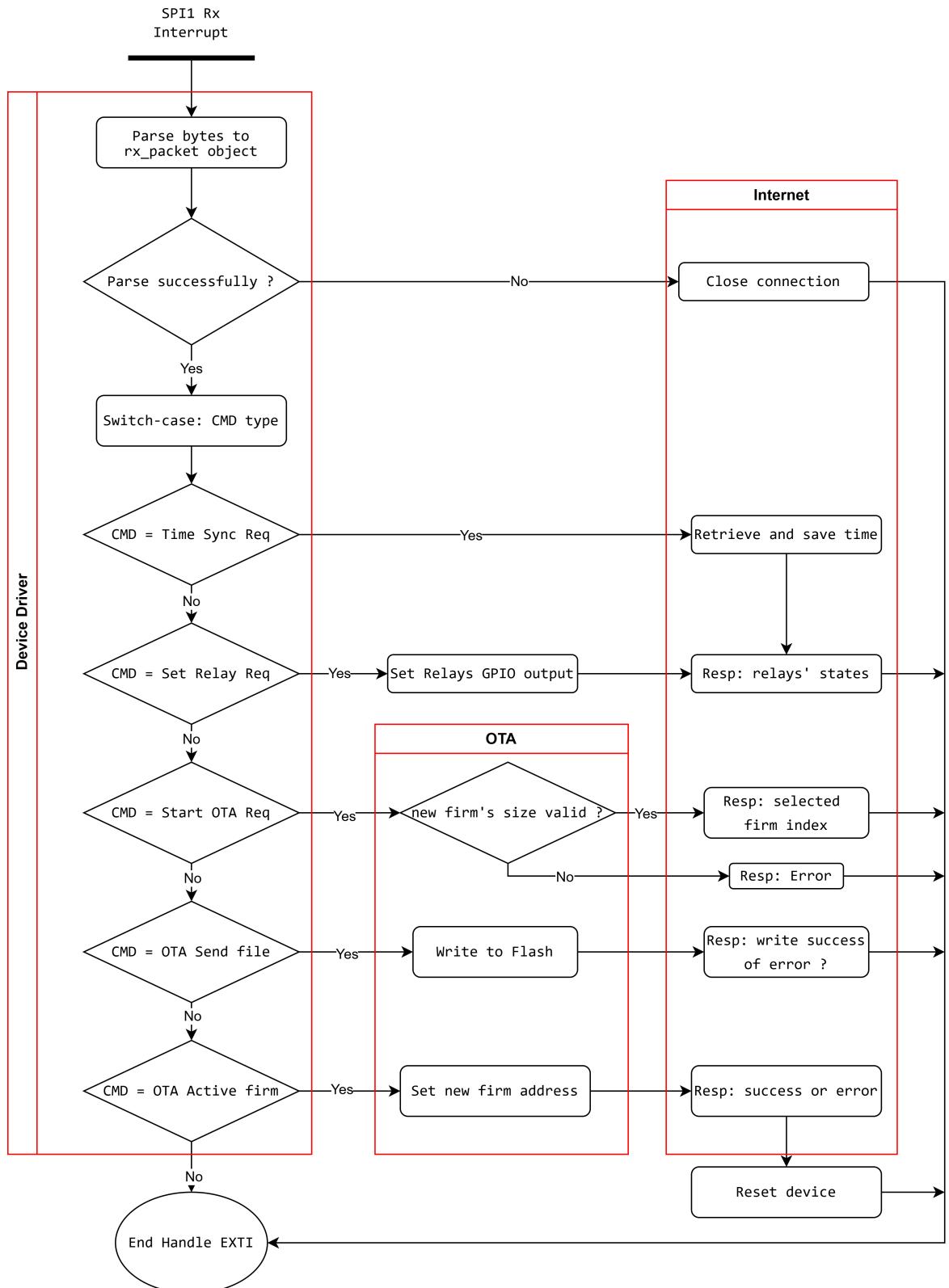
Về tổ chức các thư mục, tại thư mục core/Src tổ chức thành 4 thư mục lớn chứa 4 modules được nêu trong phần Kiến trúc triển khai firmware (xem hình minh họa ở chương trước) là: capture_driver, device_driver, internet và fota

Các thuật toán quan trọng bao gồm:

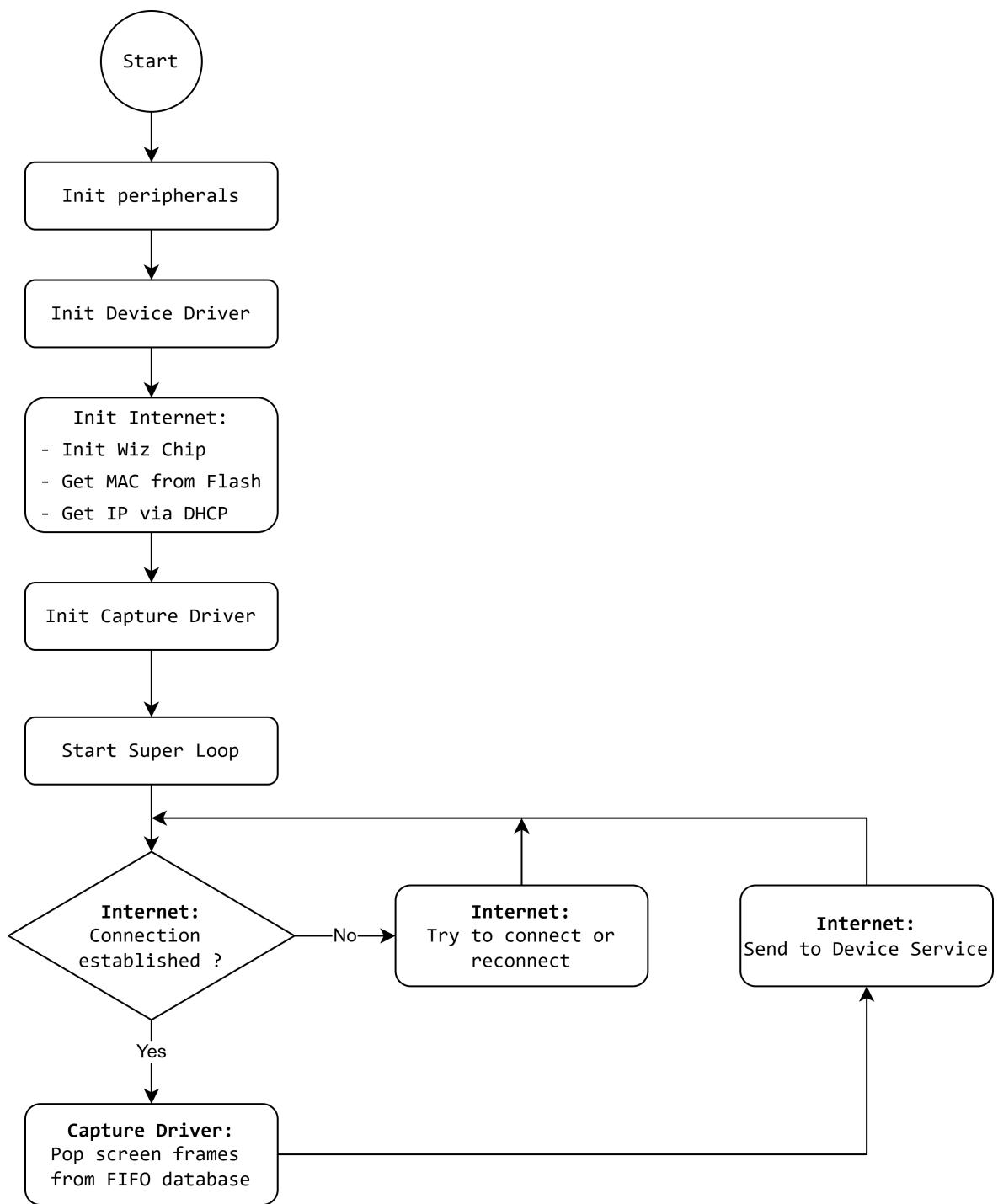
- Khởi tạo và kết nối với phần mềm Device Service
- Xử lý ngắt SPI1: Nhận yêu cầu từ Device Service
- Xử lý ngắt SPI2: Nhận SPI byte từ CPLD
- Xử lý ngắt báo Frame Stop: Báo kết thúc 1 screen frame và tiến hành lưu frame



Hình 4.6. Thuật toán xử lý ngắt nhận dữ liệu màn hình



Hình 4.7. Thuật toán xử lý ngắn dứt yêu cầu Device Service



Hình 4.8. Thuật toán cho chương trình chính

Cấu trúc gói tin giao tiếp giữa Thiết bị đọc ghi màn hình (Device) và Phần mềm giải mã (Device Service) được nêu trong bảng 4.1.

Trường	Kích thước	Offset	Mô tả
Packet Length	2 bytes	0–1	Tổng chiều dài gói tin trừ đi 12 byte đầu (độ dài Payload).
Device MAC	6 bytes	2–7	Địa chỉ MAC của thiết bị đọc ghi màn hình
CMD	1 byte	8	Loại lệnh (command) được gửi (xem bảng 4.2.).
Reserved	3 bytes	9–11	3 byte dự phòng, chưa sử dụng.
Payload	Tùy loại CMD	12 trở đi	Dữ liệu phụ thuộc vào từng loại CMD, ví dụ frame màn hình, trạng thái relay, v.v.

Bảng 4.1. Cấu trúc gói tin giữa Device và Device Service

Các loại Command (CMD) để giao tiếp với Thiết bị:

CMD	Tên	Mô tả
0x01	CMD_HANDSHAKE	CMD Handshake, dùng để gửi từ ‘Device’ đến ‘Device Service’.
0x02	CMD_REPORT_RAWDATA	CMD gửi frame màn hình, dùng để stream screen frame từ ‘Device’ đến ‘Device Service’.
0x03	CMD_REPORT_RELAY	CMD gửi trạng thái relay từ ‘Device’ đến ‘Device Service’.
0x04	CMD_SET_RELAY	CMD yêu cầu đặt trạng thái relay từ ‘Device Service’ đến ‘Device’.
0x05	CMD_TIME_SYNC	CMD yêu cầu đồng bộ thời gian từ ‘Device Service’ đến ‘Device’.
0x06	CMD_PING	CMD ping kiểm tra kết nối từ ‘Device Service’ đến ‘Device’.
0x07	CMD_OTA_START	CMD yêu cầu bắt đầu OTA từ ‘Device Service’ đến ‘Device’.

0x08	CMD_OTA_SEND_FILE	CMD yêu cầu gửi file OTA từ ‘Device Service’ đến ‘Device’.
0x09	CMD_OTA_ACTIVE_FIRM	CMD yêu cầu kích hoạt firmware mới từ ‘Device Service’ đến Device’.
0x0A	CMD_OTA_OTA_RESP	CMD phản hồi OTA từ ‘Device’ đến ‘Device Service’.

Bảng 4.2. Danh sách các loại CMD trong enum Cmd_t

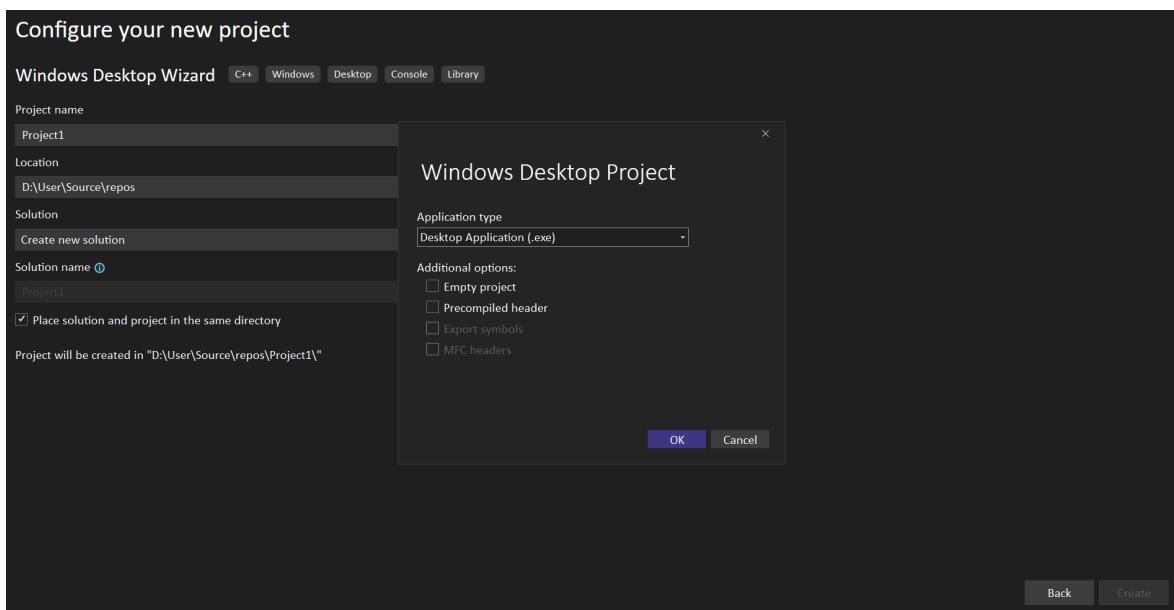
4.1.3 Phần mềm giải mã Device Service

Chuẩn bị môi trường:

- Môi trường phát triển: Visual Studio.
- Ngôn ngữ: C++.
- Loại dự án: Win32 API C++.
- Các thư viện được sử dụng: Boost,

Các bước triển khai:

Bước 1: Tạo 1 Win32 Application Project



Hình 4.9. :Tạo C++ Win32 Project trên Visual Studio

Bước 2: Triển khai chế độ chạy cho phần mềm.

Triển khai để khi phần mềm được khởi chạy sẽ:

- Tạo 1 mutex riêng có tên cố định trên Window, đảm bảo dù nhấn mở phần mềm nhiều lần thì vẫn sẽ chỉ có 1 Instance của App đang chạy.
- Phần mềm không có giao diện, sẽ là 1 tác vụ chạy ngầm trên Window (Window Service).

Bước 3: Viết các khôi triển khai.

Các khôi triển khai tuân theo thiết kế ở hình [3.12](#).

Phần này nêu chi tiết những thuật toán quan trọng cho các khôi

Thuật toán bộ giải mã và phát hiện trạng thái máy

Khôi thực hiện: Logger.

Một ‘log‘ (lượt bơm) được chốt sau khi giá trị màn hình (số lít và giá tiền) dừng tăng trong 5 giây, hoặc cột đang bơm thì dừng và reset màn hình.

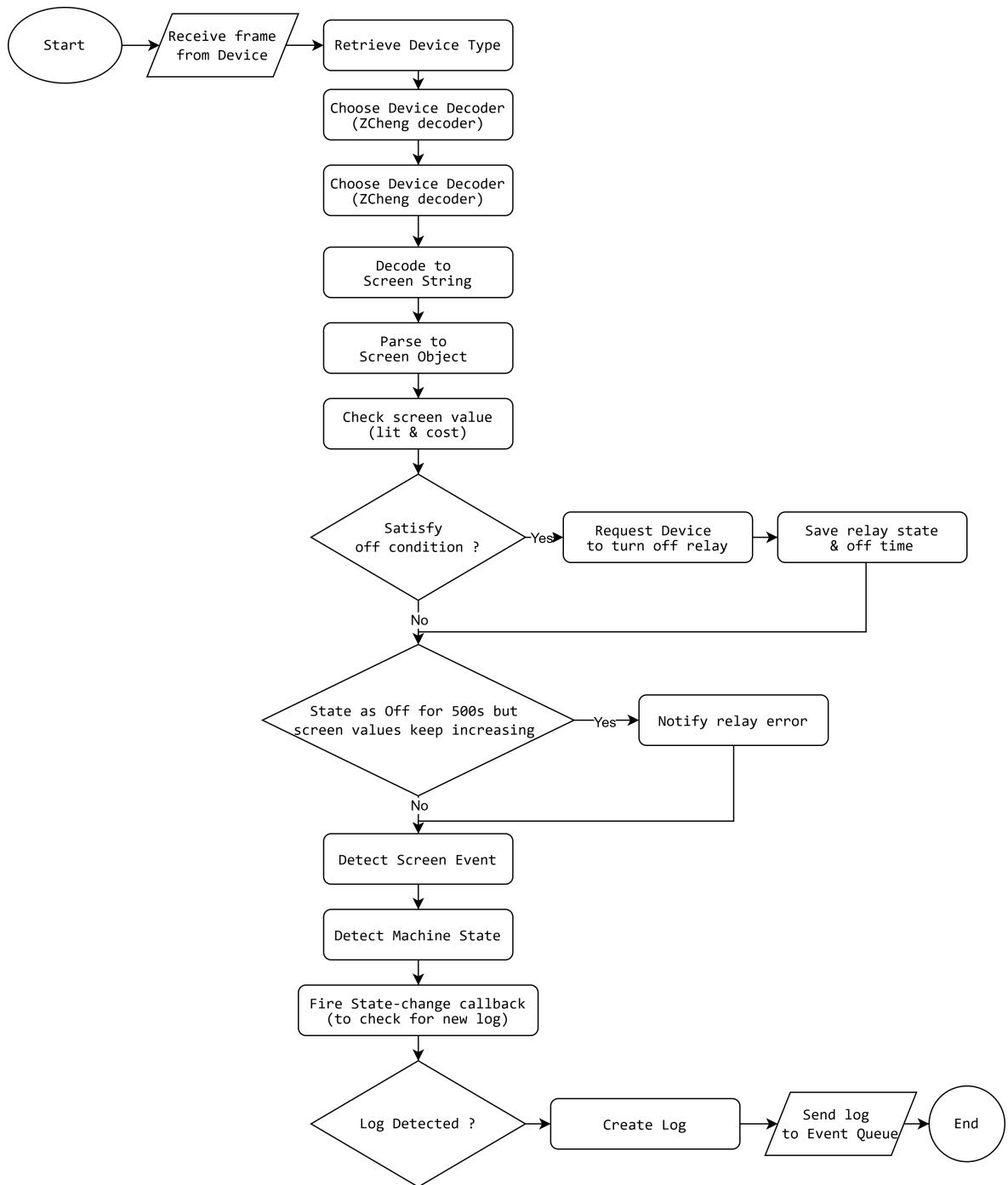
Một ‘log‘ cũng có thể dùng để lưu màn hình hiển thị giá trị preset (đặt trước lượng xăng cần bơm)

Một ‘sub log‘ cũng là một loại ‘log‘, để bổ sung thông tin cho một ‘log‘ đã có trong trường hợp việc bơm tạm dừng quá 5 giây nhưng lại được bơm tiếp thay vì reset.

Một ‘log‘ đi cùng với (các) ‘sublog‘ của nó sẽ tạo thành 1 hóa đơn hoàn chỉnh.

Trong trường hợp màn hình lỗi xảy ra, dữ liệu cũng sẽ được lưu vào 1 sublog.

Thuật toán giải mã màn hình phát hiện lượt bơm (Log Process):



Hình 4.10. Thuật toán giải mã màn hình và phát hiện lượt bom

Các Screen Event (bảng 4.4.) được xác định sau khi giải mã 1 frame màn hình, dùng để phân loại sự kiện màn hình

LogInfo	
uint8_t	logInfoType
uuid	logId
uuid	subLogId
double	cost
double	volume
double	price
double	preCost
double	preVolume
double	prePrice
uint64_t	timestamp
string	subLogContent

Hình 4.11. Các trường dữ liệu trong một Log hoặc Sublog

Event	Mô tả
EVENT_NONE	Giá trị khởi tạo khi chưa thực hiện xác định Event
EVENT_UNKNOWN	Event không xác định được
EVENT_RESET	Báo reset màn hình
EVENT_PRESSET	Báo preset (đặt trước giá trị màn hình)
EVENT_INCREASE	Giá trị màn hình (lít và tổng tiền) đang tăng
EVENT_PAUSE	Giá trị màn hình (lít và tổng tiền) đang dừng
EVENT_ERROR	Màn hình lỗi

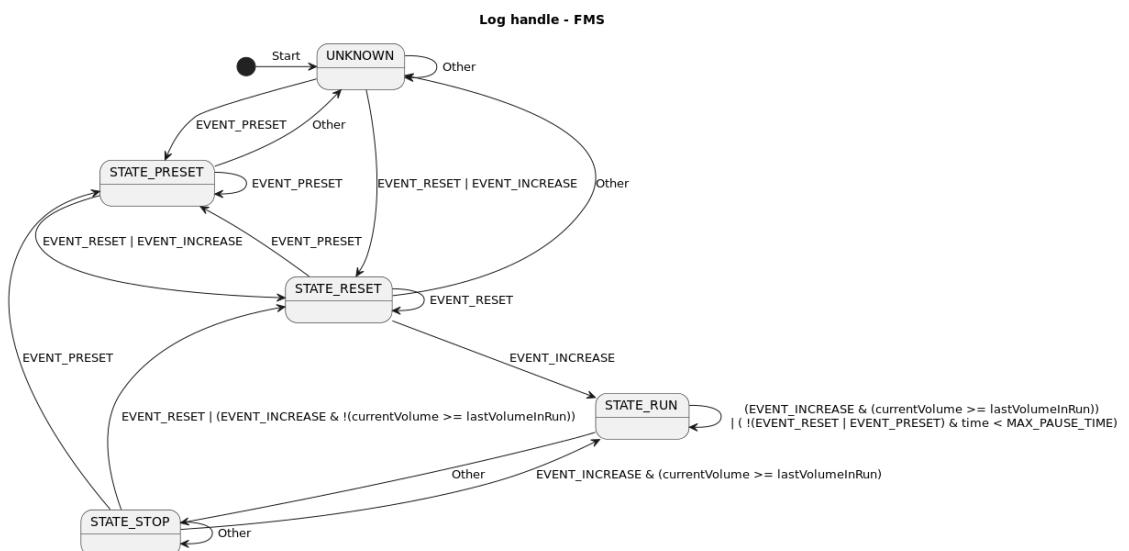
Bảng 4.3. Các loại sự kiện màn hình

Các loại trạng thái máy (Machine State, bảng 4.4.), được phát hiện dựa theo trạng thái máy trước đó kết hợp với sự kiện màn hình mới nhất.

Trạng thái	Mô tả
STATE_UNKNOWN	Không rõ, chưa xác định được trạng thái máy
STATE_RESET	Báo màn hình đã được reset (giá và thể tích đặt về 0)
STATE_PRESET	Báo người dùng đã preset (đặt trước) số lít hoặc số tiền
STATE_RUN	Đang trong quá trình bơm, giá trị trên màn hình đang tăng
STATE_STOP	Quá trình bơm đang tạm dừng, giá trị trên màn hình không tăng
STATE_ERROR	Trạng thái lỗi, phát hiện màn hình lỗi

Bảng 4.4. Mô tả các trạng thái máy (Machine State)

Việc phát hiện sự kiện màn hình và chuyển đổi trạng thái máy được chỉ ra trong sơ đồ trạng thái (hình 4.12.). Từ đó theo dõi được tình trạng cột bơm.



Hình 4.12. Sơ đồ trạng thái cho cột bơm

Sau khi xác định trạng thái mới, hàm callback tương ứng được gọi để kiểm tra phiên bơm mới nếu có:

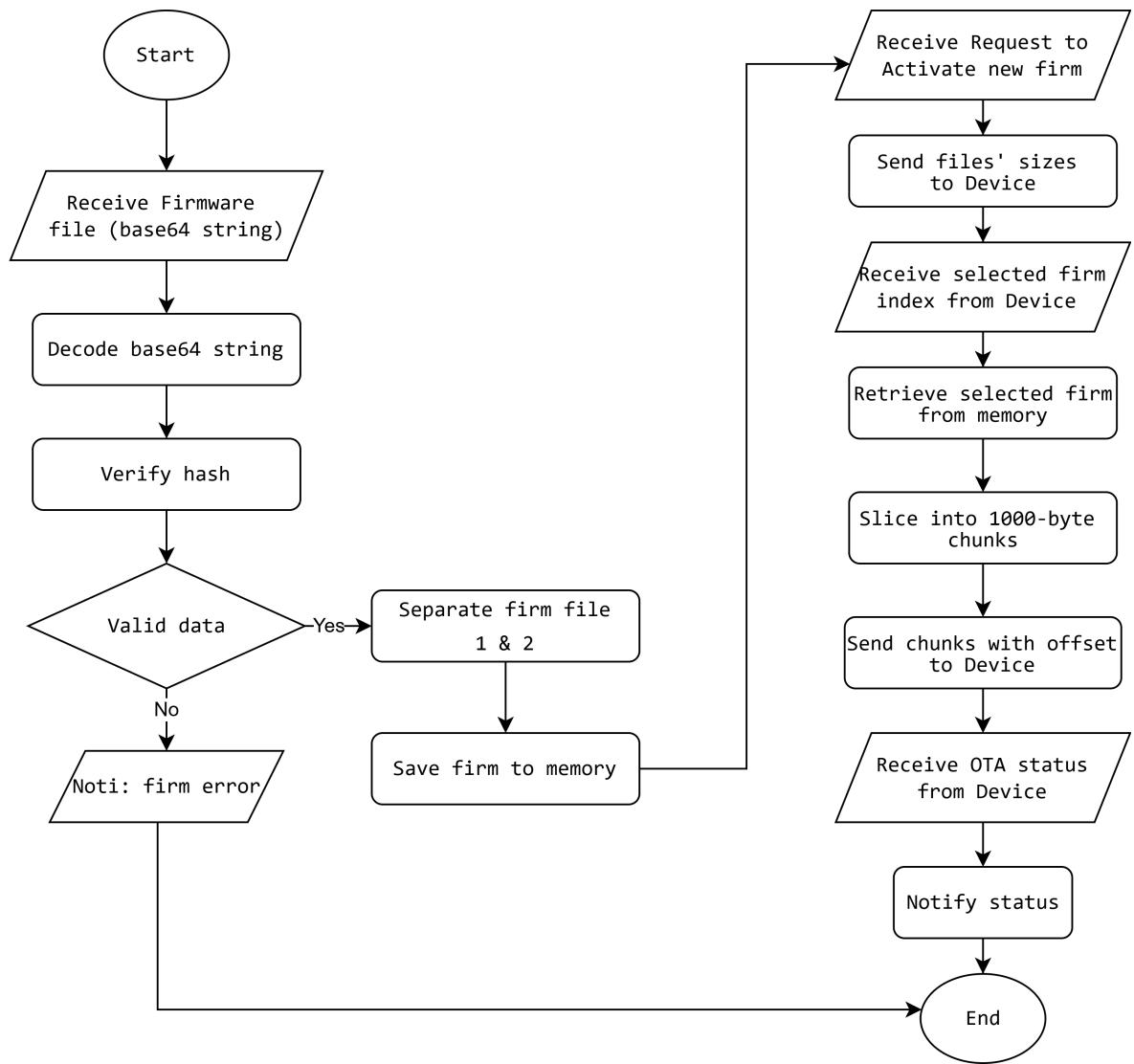
Trạng thái cũ	Trạng thái mới	Hàm callback	Mô tả
STATE_PRESET	STATE_PRESET	Check overtime to cut preset	Kiểm tra xem màn hình ở trạng thái PRESET đã được giữ hơn 5 giây chưa. Nếu có, tạo một log mới cho preset.
STATE_RESET	STATE_RUN	Generate new Log	Tạo mới một log trống để bắt đầu ghi nhận dữ liệu màn hình.
STATE_RUN	STATE_RUN	Update Log info	Trong khi vẫn ở trạng thái RUN, hệ thống cập nhật thông tin log (ví dụ: thời gian, tiền, lít, đơn giá...).
STATE_RUN	STATE_STOP	Cut Log	Khi chuyển từ RUN sang STOP, hệ thống chốt log hiện tại

Bảng 4.5. Callback được gọi sau khi xác định trạng thái mới

Các firmware file được gửi tới Device Service dạng base64-string, được kiểm tra tính hợp lệ sau đó lưu vào bộ nhớ tạm.

Khi có yêu cầu kích hoạt firmware, Device Service tiến hành:

- Hỏi Thiết bị đọc màn hình file firmware cần cập nhật
- Chia file firmware được chọn dưới dạng các chunk độ dài 1000-byte
- Gửi từng chunk tới Thiết bị đọc ghi màn hình cùng offset.



Hình 4.13. Thuật toán thực hiện OTA trên Device Service

Thuật toán thực hiện OTA

API cho Client để giao tiếp với Device Service

Device Service cung cấp API mở để bên Client UI hoặc các bên Client khác có thể giao tiếp để:

- Nhận dữ liệu thời gian thực và các phiên bơm (log), trạng thái relay
- Yêu cầu cập nhật relay hoặc thực hiện OTA.

Dạng dữ liệu: JSON.

Giao thức: TCP.

Mỗi request message là 1 JSON bao gồm các trường

- Message ID: UUID định danh message
- Message Type: Loại message để phân loại nội dung.
- Và các trường khác là payload tùy thuộc vào Message Type.

Các Message Type được hỗ trợ:

MSG Type	Mô tả
0	Báo cáo sự kiện Log/SubLog từ Device Service tới UI Client. Bao gồm thông tin log hoặc sublog được chốt trong quá trình sử dụng thiết bị.
1	Báo cáo sự kiện đặt Preset từ Device Service tới UI Client. Được gửi khi nội dung preset không đổi trong 5s hoặc khi chuyển từ preset về reset.
2	Yêu cầu từ UI Client tới Device Service để thay đổi trạng thái bật/tắt Relay và điều kiện tự động tắt Relay.
3	Device Service báo cáo trạng thái của các Relay về UI Client. Có thể là phản hồi cho yêu cầu Set Relay hoặc báo cáo chủ động khi có thay đổi.
5	UI Client gửi yêu cầu tải xuống file firmware về Device Service. File dưới dạng base64, chứa nội dung cần cập nhật cho thiết bị.
6	UI Client yêu cầu Device Service kích hoạt firmware mới đã được tải xuống. Sau khi thực hiện, thiết bị sẽ tự động reset.
7	UI Client yêu cầu Device Service trả về danh sách trạng thái hiện tại của tất cả các thiết bị đang quản lý.
8	Device Service chủ động báo cáo trạng thái của một thiết bị đọc ghi màn hình (kết nối, loại thiết bị, phiên bản firmware...) về cho UI Client.

Bảng 4.6. Danh sách các message_type của Device Service

Tạo các file log để quan sát tiến trình chạy (hình 4.14.):

- deviceInterface.txt: file log cho khối Device Interface, ghi các thiết bị đã kết nối, kèm thông báo và lỗi trong quá trình chạy.
- server-debug.txt: file log thông báo và lỗi trong quá trình kết nối và giao tiếp Client UI.
- UDPReport.txt: Thông báo hiện trạng của tiến trình broadcast UDP.

The screenshot shows a Windows desktop environment with several windows open:

- Code Editor:** Shows a C++ file named `main.cpp` with code related to command-line parsing and mutex creation.
- Log File 1:** `deviceInterface.txt` containing logs from a device interface.
- Log File 2:** `config.json` containing logs from a UDP report.
- Log File 3:** `logi-debug.txt` containing logs from a log interface.
- Log File 4:** `server-debug.txt` containing logs from a server port.
- File Explorer:** Shows project files including `DeviceInterface.cpp`, `DeviceInterface.h`, `main.cpp`, `utils.h`, and `utils.cpp`.
- Output Window:** Shows standard output and error messages from the build process.

Hình 4.14. Device Service: Các file logs

4.1.4 Viết phần mềm Client UI hiển thị các phiên bom

- Môi trường sử dụng: NodeJS v20.18.1
- Ngôn ngữ: TypeScript
- Database: SQLite
- Các thư viện chính sử dụng: ReactJS, sqlite3, MUI Components

Các bước triển khai:

Bước 1: Tạo cấu trúc dự án Electron

- Sử dụng câu lệnh `npm create electron-vite@latest` để setup dự án.
- Chọn template là ReactJS, ngôn ngữ TypeScript, nhấn enter để bắt đầu tạo mẫu.

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows the project structure under "DEVICE-SERVICE-CLIENT".
- Editor:** Displays the content of "index.ts" which contains code for creating a BrowserWindow.
- Status Bar:** Shows "Duo" mode, file paths ("package.json M", "index.ts M"), and other system information.

Hình 4.15. Mẫu dự án sử dụng Electron + Vite + ReactJS

Bước 2: Triển khai các tác vụ ở Main Process bao gồm:

- Tạo 2 TCP Client:
 - Realtime Screen Device Service Client: để nghe dữ liệu màn hình thời gian thực, chuyển tiếp màn hình tới Render Process ở cổng 5001
 - Event Device Service Client: để nhận phiên bơm (log), điều khiển relay ở cổng 5002. Các phiên bơm nhận được sẽ được chuyển tiếp tới Database ORM.
- Tạo Database ORM để đọc ghi cơ sở dữ liệu phiên bơm.

Cơ sở dữ liệu phiên bơm gồm 1 bảng lưu các log/sublog gồm các trường:

LogInfo	
uint8_t	logInfoType
uuid	logId
uuid	subLogId
double	cost
double	volume
double	price
double	preCost
double	preVolume
double	prePrice
uint64_t	timestamp
string	subLogContent

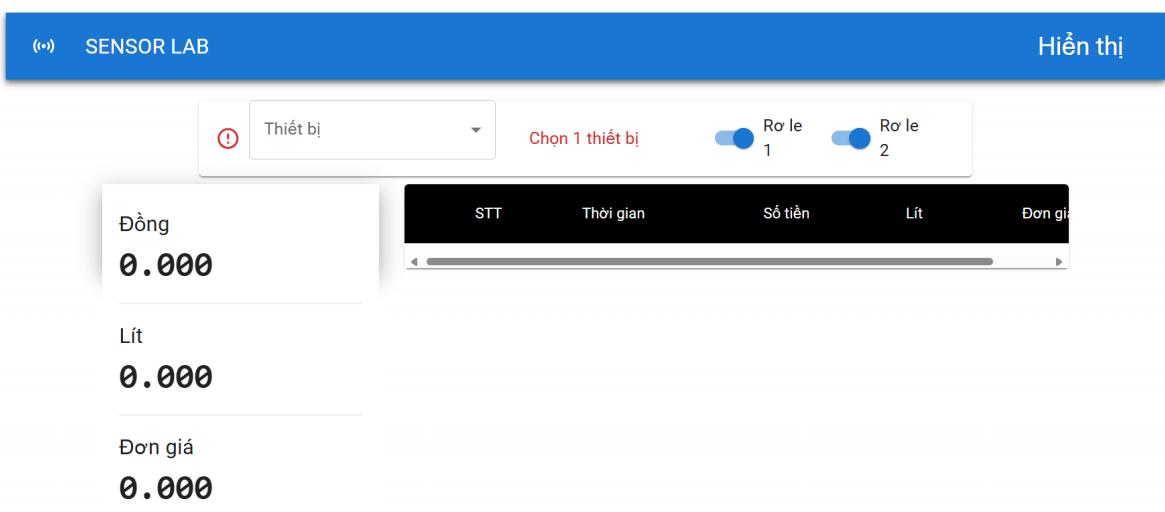
Hình 4.16. Các trường lưu trong bảng Log của Database

Bước 3: Triển khai giao diện tại Renderer Process

Sử dụng MUI để xây dựng các component:

- Khung hiển thị màn hình thiết bị (3 dòng).
- Bảng thống kê log/sublog theo thời gian thực.
- Lọc, tìm kiếm theo device_id, log_id, thời gian, trạng thái, ...
- Hiển thị trạng thái kết nối và relay theo từng thiết bị. Các nút điều khiển relay

Giao diện hoàn chỉnh



Hình 4.17. Giao diện hiển thị Client UI

KẾT LUẬN

TÀI LIỆU THAM KHẢO

PHỤ LỤC

A Một số phương pháp đo và hiệu chuẩn

Phụ lục cần thêm (nếu có) ...